SANDIA REPORT

SAND2004-2129
Unlimited Release
Printed June 2004

ROCIT: A Visual Object Recognition Algorithm Based on a Rank-Order Coding Scheme

Paul C. Reeves, Benjamin D. Farkas, John J. Jones, and Antonio I. Gonzales

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore California 94550

Sandia is a multiprogram laboratory operated by Sandia Corporation,
a Lockheed Martin Company, for the United States Department of Energy's
National Nuclear Security Administration under Contract DE-AC04-94AL85000.

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865)576-8401
Facsimile: (865)576-5728
E-Mail: reports@adonis.osti.gov
Online ordering: http://www.doe.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd
Springfield, VA 22161

Telephone: (800)553-6847
Facsimile: (703)605-6900
E-Mail: orders@ntis.fedworld.gov
Online order: http://www.ntis.gov/help/ordermethods.asp?loc=7-4-0#online

SAND2004-2129
Unlimited Release
Printed June 2004

ROCIT: A Visual Object Recognition Algorithm Based on a Rank-Order Coding Scheme

Paul C. Reeves, Benjamin D. Farkas,
John J. Jones, and Antonio I. Gonzales
Next Generation Monitoring Systems
Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185-1138

Abstract

This document describes ROCIT, a neural-inspired object recognition algorithm based on a rank-order coding scheme that uses a light-weight neuron model. ROCIT coarsely simulates a subset of the human ventral visual stream from the retina through the inferior temporal cortex. It was designed to provide an extensible baseline from which to improve the fidelity of the ventral stream model and explore the engineering potential of rank-order coding with respect to object recognition. This report describes the baseline algorithm, the model's neural network architecture, the theoretical basis for the approach, and reviews the history of similar implementations. Illustrative results are used to clarify algorithm details. A formal benchmark to the 1998 FERET fafc test shows above average performance, which is encouraging. The report concludes with a brief review of potential algorithmic extensions for obtaining scale and rotational invariance.

Acknowledgements

This work was performed under the Next Generation Intelligent Systems Grand Challenge LDRD program of Sandia National Laboratories. Special thanks go to the project managers who have supported this work as a component of the larger theme, namely Larry Ellis, Russ Skocypec, David Gallegos, and John Wagner. Additional thanks also go to John Ganter and Kurt Larson who have served as project leads for the Perceptive Systems team. Kurt Larson and Antonio Gonzales also provided valuable technical reviews. Their thoughtful comments have improved the document and are greatly appreciated.

This work was supported by the United States Department of Energy under Contract DE-AC04-94AL85000. Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy.

Contents

This page intentionally left blank.

Figures

Figures (continued)

Tables

1. Introduction

Object recognition in cluttered, uncontrolled environments is a fundamental need across myriad problem domains, from the mundane to critical national security applications. Competent computational assistance with this task would be welcome, especially in light of expanding acquisition of imaging data. Consequently, object recognition is an outstanding problem in computer science and an active area of research in the computer vision, digital image processing, pattern recognition, and computational neuroscience disciplines.

Biomimetic approaches to engineering and computation have proven effective for some problems, for example, biomimetic engineered materials, genetic/evolutionary optimization algorithms, and neural networks for pattern recognition and classification. They are appealing because there is usually a natural existence proof for justification, and if the natural analogue can be understood, then it can be exploited as a technical road map. The remarkable vision system common to humans and primates is an example of how object recognition can be accomplished in nature, and critical understanding of this system has reached a threshold level suitable for developing computational models. The work presented here takes inspiration from this foundational knowledge for the purposes of developing computational object recognition system.

This SAND report documents ROCIT, a MATLAB® implementation of a neural-inspired object recognition algorithm. ROCIT uses the Rank-Order neural Coding scheme of *Gautrais and Thorpe* [1998], a new and novel neuroscience hypothesis, to simulate the ventral pathway of the human vision system that extends from the retinae to the Inferior Temporal cortex (IT). This coding scheme uses the latency of the first spike in a neuron's spike train as a way of encoding visual information. This is in contrast to temporal or frequency coding schemes, and attempts to explain the rapid activation times (80-120ms) reported for cells in the inferior temporal cortex where cells implicated in object recognition are found (See, for example, *Logothetis and Sheinberg* [1996]; *Tanaka* [1996]; *Thorpe et al*. [1996]).

ROCIT is a prototype application designed to enable rapid algorithm exploration and development. A faster C++ implementation, SpikeWave, has also been developed for the purpose of running computationally-intensive benchmarks. Both implementations share a common baseline algorithm as described in this report.

This page intentionally left blank.

2. Background

Humans and primates share a similar neuroanatomy for visual processing. Intensive study over the past several decades has led to a general consensus as to the basic function of this system [*Hubel and Weisel*, 1977][1], although fundamental questions remain. In general, the visual system consists of a hierarchical arrangement of neuron groups having increasingly complex preferred stimuli and receptive field sizes[2] as the synaptic distance from the eyes increases.

2.1 Visual Streams

Visual information enters through the pupils and is transformed into neural impulses by the neuronal layers that comprise the retinae, the complex sheets of neural cells forming the posterior wall of the posterior chamber of the eye. The transformation begins with light sensitive rod and cone cells. Cones are sensitive to color but are restricted primarily to the region of the fovea, a small area of very high cell density at the centers of the retinae. Rods, on the other hand, are found throughout the retinae. They are color insensitive and have a much larger dynamic range than cone cells with respect to illumination. Rods and cones form synaptic connections with several cell types in the retinae, the output of which are approximately 1,000,000 retinal ganglion cells per eye whose axons form the optic nerve.

The spatial resolution (*i.e.*, receptive field) of retinal ganglion cells at the fovea is about 0.03º and decreases to nearly 3º at the periphery. Of the neurons that make up the retinae, only retinal ganglion cells produce action potentials (a.k.a., "spikes" in membrane voltage), whose temporal spike pattern presumably encodes visual information. The optic nerves project to the brain stem where they synapse onto various projections, primarily the lateral geniculate nucleus (LGN) (See Figure 2-1.). Within the LGN, connections originating from retinal rod and cone cells are segregated into distinct layers. These are referred to as the magnocellular and parvocellular pathways, respectively. Since rods and cones respond to different stimuli (black and white vs. color) and illumination contrast, as well as having differing degrees of spatial resolution and response times, this segregation results in an early stage of information routing that is used for specialized processing in later stages.

From the LGN the optic radiations project to the occipital cortex in the posterior of the brain, an area termed the primary visual cortex or V1. It is a thin (~2 mm thick) cortical area consisting of six distinct layers and having about 200,000,000 neurons. This is the first of several functionally-defined brain regions referred to as visual areas. Visual areas are defined by their retinotopic[3] cell arrangement and neuronal stimulus preferences. There are

---

[1] In 1981, David H. Hubel and Torsten N. Wiesel were awarded the Nobel Prize in Physiology or Medicine for their work on the neuroscience of vision.

[2] A neuron's receptive field is the portion of the visual field that a neuron will respond to. The level of response, as defined by spike rate or latency, depends on the similarity between a cell's preferred stimulus and the presented stimulus.

[3] Retinotopic refers to a neuron arrangement where adjacent neurons respond to adjacent regions of the visual field. The visual areas are, in part, defined by possessing retinotopically-arranged neuron layers.

several distinct cell types in V1. Simple cells respond to oriented edges, line segments, and end-stopped lines. Complex cells in V1 respond to moving lines and edges, while "blob" cells respond to color patches. Simple and complex cells receive input from the magnocellular pathway while blob cells receive input from the parvocellular pathway, although the cell types have extensive interconnections. *Raizada and Grossberg* [2003] and *Grossberg* [2003] provide and excellent reviews of synaptic connections in the cortex together with an interpretation of their functional roles.

Beyond V1, two functionally-distinct pathways of visual processing have been identified: the dorsal stream and the ventral stream (See, for example, *Ungerleider and Haxby* [1994].). The dorsal stream, also referred to as the "where" stream, is responsible for sensing types of motion relative to the viewer. The dorsal stream receives the majority of its input from the magnocellular pathway via a feed forward circuit through the visual areas V1, V2, V3, and the Middle Temporal Area (a.k.a., MT or V5) before terminating in the posterior parietal cortex. The ventral stream, also known as the "what" stream, is responsible for object recognition and terminates in the inferotemporal cortex (a.k.a., the inferior temporal cortex or IT). It receives most of its input from the parvocellular pathway in a feed forward circuit through V1, V2, and V4 terminating in the IT.

Figure 2-1 shows a simplified version of these processing pathways. While the dorsal and ventral streams are usually described in terms of their forward connections, the actual synaptic connections of the visual areas are far more complex. There are extensive feedback and lateral connections, some of which are shown. The roles played by the various connections are not fully understood. *Lamme and Roelfsema* [2000] provide an excellent review of current scientific thoughts on this topic, while *Raizada and Grossberg* [2003] review ideas on the detailed connections between and within the LGN, V1, and V2.[4]

2.2 Object Recognition in the IT

The IT is not a visual area, by definition, since it no longer contains retinotopically–arranged neurons. Within it are spatially-distinct areas of neurons that are selective to specific types of objects or object primitives. Direct neural recording and functional magnetic resonance imaging (fMRI) in primates[5] and humans have been used to delineate these regions based on their preferred stimuli. In humans there are groups of cells selective to physical places (parahippocampal place area), human faces (fusiform face area), body parts (extrastriate

[4] It is also worth noting that there are many other visual areas whose roles and involvement with the ventral and dorsal streams are not yet clear. See *Felleman and Van Essen* [1991], and/or "Functional Brain Areas in the Human" at: http://defiant.ssc.uwo.ca/Jody_web/fMRI4Dummies/functional_brain_areas.htm#retinotopic%20and%20visual%20areas (Accessed April, 2004).

[5] The posterior and anterior IT in humans correspond roughly to the anterior inferotemporal cortical area (TE) and the temporal-occipital area (TEO) of the primate brain, respectively.

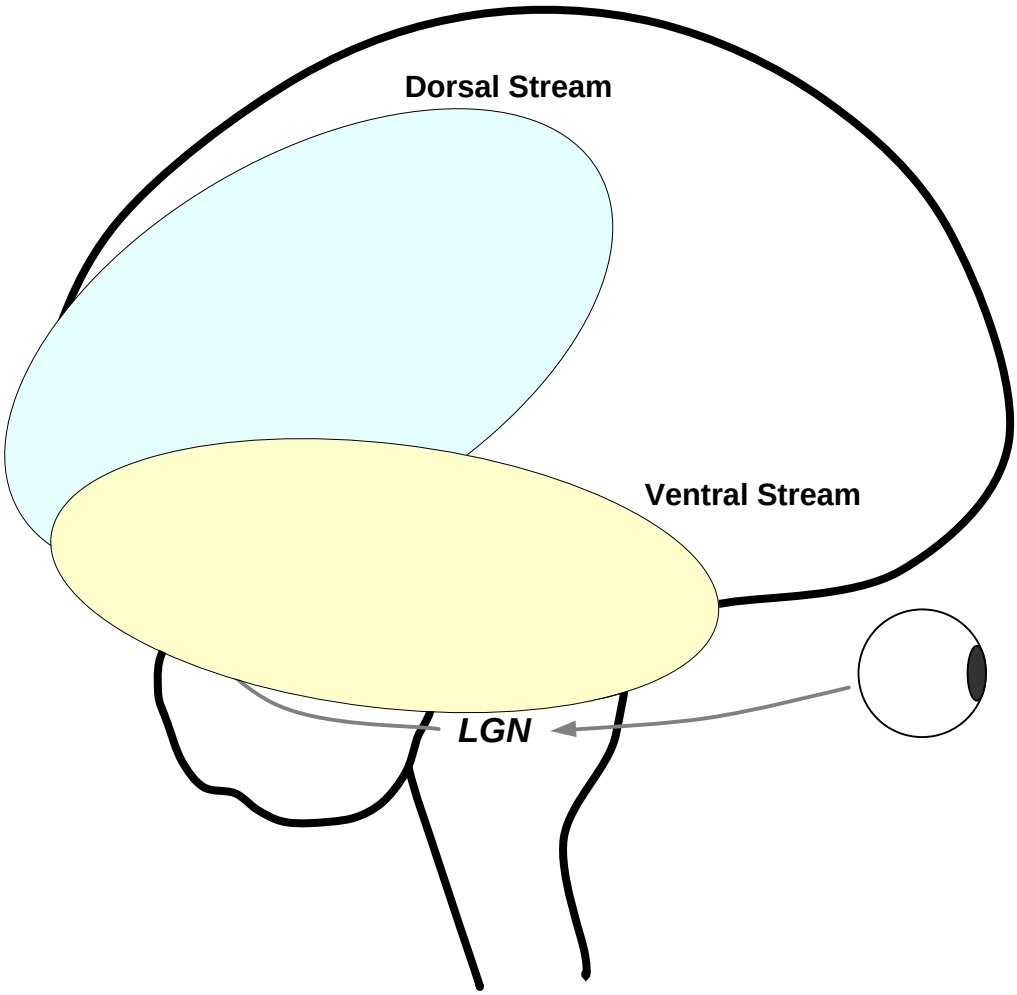


Figure 2-1. Primary visual pathways in the human vision system. The optic nerves project from the eyes to the lateral geniculate nucleus. From the LGN the optic radiation proceeds to the primary visual cortex (V1). Beyond V1 the dorsal stream leads to the parietal cortex whereas the ventral stream terminates in the inferotemporal cortex.

body area), and text (visual word form area), among others. Another area, the lateral occipital complex, seems to respond to lower-level object features (primitives) and may be used as a population code from which to build representations of novel objects. Excellent reviews of this work, together with foundational insights can be found in *Grill-Spector et al*. [1999; 2001], and *Grill-Spector* [2003]. Neurons of the IT that respond to complete objects

are selective to specific views and illumination conditions of an object, but have invariant responses with respect to object position, scale, and representation (*e.g*., a line drawing or silhouette vs. a color photograph). These results bolster the theoretical perspective that object recognition is assembled from a set of 2D views (*e.g*., *Poggio and Edelman* [1990], *Edelman* [1999]), rather than from 3D object-centered representations (*e.g.*, *Biederman* 1987]).

This interpretation is strengthened by neuronal recordings from primates. *Tanaka et al*. [1991] found that cells in the TEd (dorsal TE) are sensitive to orientation[6] with responses dropping by more than 50% with rotations of +/- 90º. *Ito et al*. [1991] found variability in responses to objects of different sizes, but with considerable ranges of scale invariance. Some cells had stable responses with four octaves of scale change, while others showed response drop-off after two octaves of scale. *Tanaka* [1996] provides and excellent review of the state of knowledge of the primate IT at the time. N.K. Logothetis and co-workers [*Logothetis et al*., 1994; *Logothetis and Pauls*, 1995; *Logothetis et al*., 1995] measured sensitivity to object viewpoints and found that cells were tuned to a specific viewpoint with invariance for out of plane rotations to about +- 40º. They found that macaques could learn to recognize an object at any rotation given as few as three training views spaced at 120º rotation.

Our brains provide a seamless percept of objects that is invariant to changes in translation (position in the visual field), scale, rotation, and representation even when the object is embedded in a cluttered scene, when lighting conditions vary, or when the object is wholly novel. The fundamental challenge is to understand how a set of view-specific object/feature-tuned cells can be coupled to accomplish this miraculous feat.

2.3 Modeling Approaches to Object Recognition

This foundational understanding has led researchers to develop variety of computational models of the visual system. These include the Neocognitron [*Fukishima*, 1980; *Fukushima and Miyake*, 1982], the adaptive resonance theory based networks of Stephen Grossberg [*Carpenter and Grossberg* 1987a, 1987b; *Bradski and Grossberg*, 1995], VisNet/VisNet2 [*Wallis and Rolls*, 1997; *Rolls and Milward*, 2000], HMAX [*Riesenhuber and Poggio*, 1999], and SpikeNET [*Delorme et al*., 1999], among others.

The primary focus has been on object recognition rather than motion, and the dominant motivation has been to validate our theoretical understanding through modeling. All use a neural network composed of hierarchical neuron layers representing the visual areas along the ventral stream of the human/primate vision system with differing degrees of fidelity, and none can be considered a complete representation. With the exception of SpikeNET, all use a frequency code to model neural spike trains. SpikeNET uses the Rank-Order Coding (ROC) scheme developed by *Gautrais and Thorpe* [1998] and represents a recent break from this modeling tradition.

Figure 2-2 depicts a generic spike train and is typical of those measured in retinal ganglion cells and pyramidal cells of the visual cortex. As can be seen, there is a delay (a.k.a. "latency") between the presentation of the stimulus and the first neuron spike. This latency increases to approximately 120-150 milliseconds at the IT where recognition is believed to take place. A frequency code perspective interprets the spike frequency as an encoding of information. A time window is required over which a frequency can be computed since spike frequency exists only for multiple action potentials, but this implies that a delay is

[6] Orientation is the rotation of an object within the viewing plane.

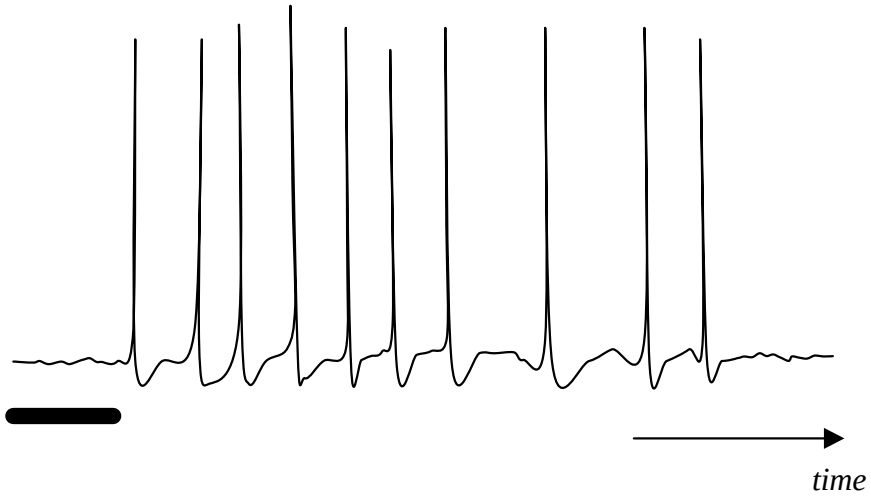


Figure 2-2. Notional schematic of a typical spike train of action potentials from a retinal ganglion cell. The dark bar indicates the presentation of visual stimulus and time progresses to the right. Vertical and horizontal scales are not defined.

required before the computation could be complete. It is difficult to reconcile frequency coding with the short activation latencies measured in humans and primates along the ventral stream (*e.g.*, *Oram and Perrett* [1992]; *Logothetis and Sheinberg* [1996]; *Tanaka* [1996]; *Thorpe et al*. [1996]; *Li et al*. [2002]. See also *Wyeth* [1999] and *Thorpe et al*. [2001] for reviews.).

Rank-order coding [*Gautrais and Thorpe*, 1998] operates on the latency of the *first* spike in a spike train. This helps explain the short latencies observed in the primate/human visual streams. It also provides a mechanism of expediting the most "salient" information in a scene [*Van Rullen and Thorpe*, 2001] where saliency is defined as the optimal stimulus of a neuron group at any stage of the ventral stream [*Van Rullen*, 2003]. ROC *does not* explain the function of the remainder of the spike train and there are severe problems in defining "first" in real systems where time is continuous, but ROC is attractive from an engineering perspective for systems deriving their input from static images.

We are aware of two previous ROC scheme implementations for visual object recognition. The first implementation was by Simon Thorpe et al. who implemented a model they named SpikeNET [*Delorme et al*., 1999]. SpikeNET uses a three-layer architecture and an event driven algorithm for propagating spikes through the system. The first layer can be interpreted as the output of the rod cell portion of the retinal circuit. The second layer consists of orientation selective cells that correspond to the simple cells of V1. These feed object selective cells that conceivably correspond to some area in the IT. Raul Muresan has implemented a very similar model, RetinotopicNET, with minor extensions [*Muresan*, 2002a; *Muresan*, 2003]. The key extension is the implementation of competition between layers of different scales in order to achieve a degree of scale independence [*Muresan*, 2002b].

This page intentionally left blank.

3. Algorithm Description

ROCIT's algorithm is similar to that of SpikeNET (See *Delorme et al*. [1999]; *Delorme and Thorpe*, [2001a, 2001b]; *Thorpe et al*. [2001] for algorithm details.), but with a few exceptions that are noted below. It uses a simple network of retinotopically-arranged, rectangular neuron layers. The neuron model is so lightweight as to have no explicit governing equation. Likewise, only the first spike of a neuron is propagated to efferent neurons so that the remainder of the spike train is not explicitly modeled and is ignored. There are two distinct modes of operation: training and recognition. These algorithmic details are described more fully in the following sections.

3.1 Network Architecture

ROCIT uses the same layered architecture as SpikeNET [*Delorme and Thorpe*, 2001a]. It consists three sets of neuron layers: a set of input layers loosely representing the retina (a.k.a. on/off layers), a set of orientation layers with neurons selective to oriented edges or lines similar to simple cells of V1, and a set of recognition layers, one for each target[7] (Figure 3-1). Because target layers have a retinotopic neuron arrangement and are only two synapses from the input, they cannot be seen as representing cells in the inferior temporal cortex. However, they are the locus of recognition in this modeling scheme. Input layers occur in pairs (an "on" and an "off") while orientation layers occur in sets of eight. During training there is a single target layer, while in recognition mode there may be any number of targets each with a corresponding target layer.

Within ROCIT each layer shares the same dimensions as the input image, *i.e*., there is a neuron for each pixel in the image. This is a different implementation than SpikeNET, which halves the number of neurons in orientation layers and again in the target layers so that target layers have one fourth the number of neurons as the input image has pixels. There is no explicit neurological basis for the coarsening used in SpikeNET, and it causes a loss of detail in target recognition. For these reasons, ROCIT does not follow SpikeNET's example for this algorithmic detail.

Each layer has an associated kernel that defines its preferred stimulus. A kernel is a rectangular array of floating point values, also referred to as a weight matrix or selectivity matrix. Every neuron in a given layer shares the kernel matrix with all other neurons in the layer. The kernels serve as synaptic weights and define the pattern of connections between afferent (upstream) and efferent (downstream) layers relative to a firing neuron. Figure 3-2 shows how the selectivity kernel defines synaptic connections between an afferent and efferent layer. By sharing kernels and using them to define a relative pattern of synapses, memory usage is minimized and computational efficiencies are achieved. The kernels are

[7] *Delorme et al*. [1999] describe assembling a target layer from intermediate features. For example, they have eye, nose, and mouth recognition layers that are used as input to a fourth layer that recognizes faces. ROCIT could easily be reconfigured to use a more complex recognition scheme that takes advantage of smaller-scale intermediate features.

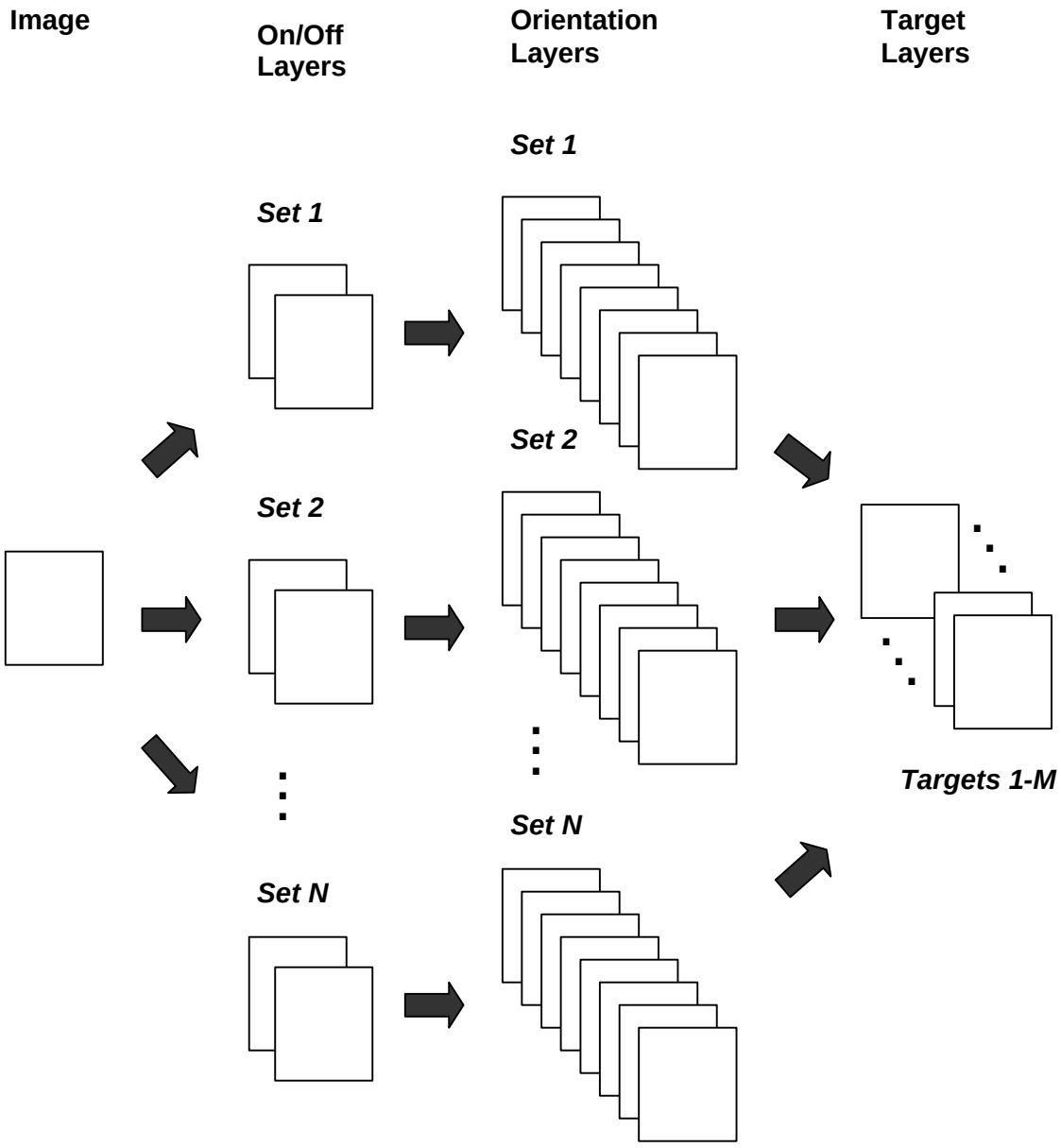
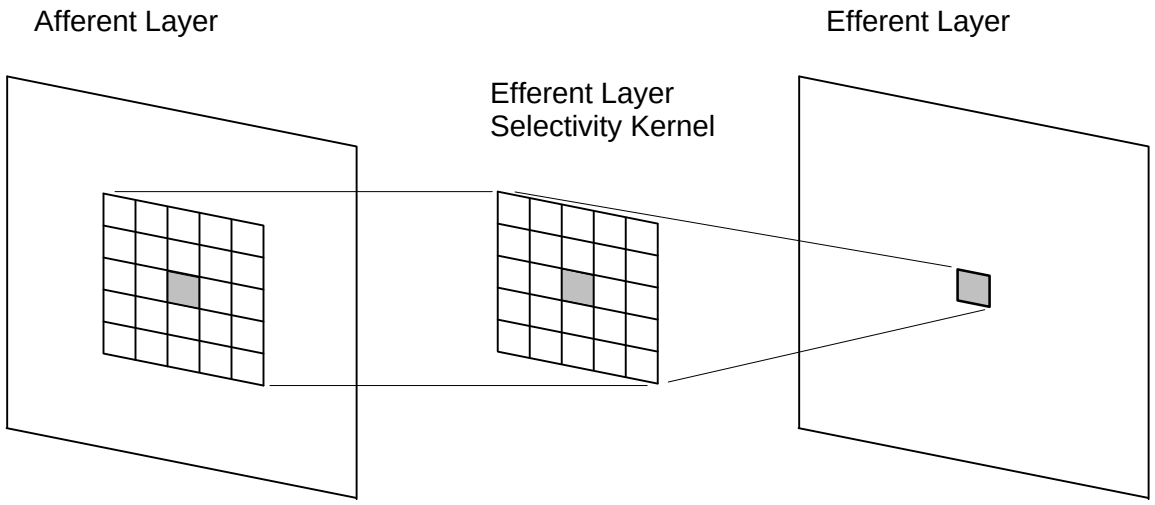
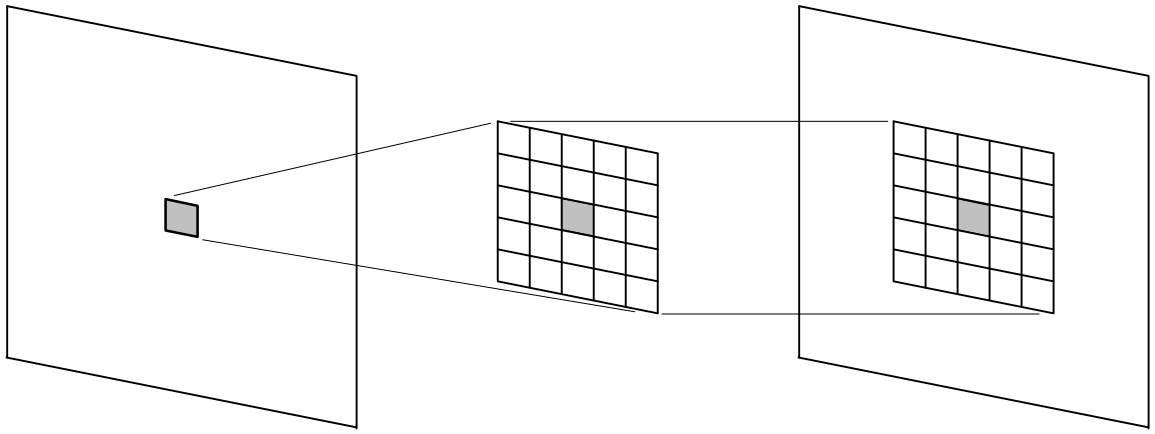


Figure 3-1. Layered architecture used by ROCIT. An image is presented to the on/off layers of the network. On/off layers occur in pairs of two (one pair for each kernel size/scale) and any number of on/off layer .sets may be used. On/off layers feed forward to sets of orientation layers. On/off layer sets of a given scale are coupled with orientation layer sets that always occur in groups of 8 orientations at 45º increments. Orientation layers feed forward to target layers. During training there is a single target layer, while in recognition mode there may be any number of targets.



(a) Region of an afferent layer that can affect an efferent neuron relative to its selectivity kernel. This can be interpreted as the receptive field of an efferent layer.



(b) Region of the efferent layer that can be affected by a particular neuron in an afferent layer relative to the efferent layer's selectivity kernel.

Figure 3-2. Relationship between an efferent layer's selectivity kernel and the synaptic pattern between afferent and efferent layers. The synaptic pattern is shared by all neurons in the layers, as is the associated weight matrix. The gray neuron aligns across the layers and kernel.

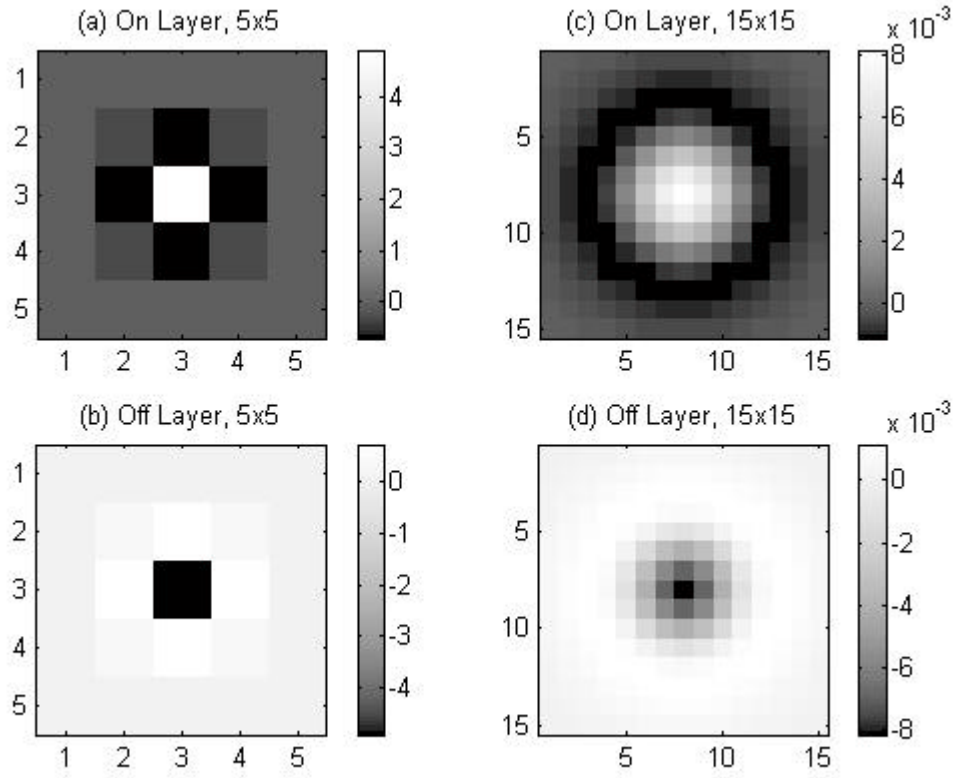


Figure 3-3. Examples of discrete Laplacian of Gaussian kernels for on/off layers: (a) On Layer, 5x5, s=0.5, (b) Off Layer15x15, s=0.5, (c) On Layer, 15x15, s=2.5, (d) Off Layer, 15x15, s=2.5.

used during training and recognition to update voltage and sensitivity values in efferent layers.

3.1.1 On/Off Layer Kernels

On and off layers have Laplacian of Gaussian (LoG) kernels. These are defined by

$$k_{LoG}(x,y) = (-1)^n \frac{1}{ps^4}\left[1-\frac{x^2+y^2}{2s^2}\right]\exp\left(-\frac{x^2+y^2}{2s^2}\right) \quad (1)$$

where s is a parameter that defines the scale of Gaussian smoothing applied, and where $n \in \{1,2\}$. The LoG operator approximates the second spatial derivative of an image and the sum of values of a discrete version is zero. Therefore, it is sensitive to edges in an image, while providing a zero response where image intensity is constant or changing linearly. On/off layers are used in pairs having a common scale parameter (s). ROCIT may have any number of on/off pairs, while SpikeNET typically uses three scales [*Delorme et al*., 1999]. As well, SpikeNET uses difference of Gaussian (doG) kernels [*Delorme et al*., 1999], which are an approximation to LoG kernels. On layers are defined by $n = 2$, giving them a positive

circular center with a negative surround. Off layers use $n = 1$, making them the inverse of on layers. This function is commonly referred to as a "Mexican hat" function because of its shape. Convolving the input image with a LoG kernel accentuates regions of bright values with dim surrounds and vice versa. This mimics the selectivity of retinal ganglion cells.

Examples of discrete on/off layer kernels are shown in Figure 3-3. The kernels are seen to be radially symmetric, to possess a surround region having the opposite sign of the center, and to decay to zero beyond the surround.

3.1.2 Orientation Layer Kernels

Orientation layers are designed to be selective to oriented line segments or edges mimicking simple cells in the visual cortex. This is accomplished via Gabor kernels. A Gabor kernel is defined by

$$k_{Gabor}(x,y) = \frac{1}{2p s_x s_y} \exp\left[-\frac{1}{2}\left(\frac{x^2}{s_x^2} + \frac{y^2}{s_y^2}\right)\right] \cdot \cos(2p u_0 x + f) \tag{2}$$

where s_x and s_y are Gaussian envelopes in the x and y directions, u_0 is the frequency of a sinusoidal plane wave along the x-axis, and f is a phase offset. This represents an oriented planar wave having an exponential decay with distance from the origin. A set of Gabor filters at other rotations and scales can be obtained through the generating function

$$f'_{mn}(x,y) = k^{-m} f_{Gabor}(k^{-m}x', k^{-m}y'), \quad k \geq 1 \tag{3}$$

where m and n are integers such that $m = 0,1,\ldots,M-1$ and $n = 0,1,\ldots,N-1$, respectively, and where *M* and *N* are the total number of scales and rotations, respectively. The parameters *k* and *m* are used for scaling, while a transformation is applied to the coordinate axes to rotate the kernel:

$$x' = x\cos q + y\sin q \tag{4}$$

$$y' = -x\sin q + y\cos q \tag{5}$$

with $q = 2pn/N$.

ROCIT uses orientation layers in sets of eight at 45º rotations, a symmetric Gaussian envelope $(s_x = s_y)$, and all layers share common parameters with the exception of their rotation angles.

An edge selective kernel arises from a phase angle (f) of 90º, while a phase angle of 0º yields line detection cells. Figure 3-4 shows examples of several discrete orientation layer kernels. Note that all kernels have been normalized to have a maximum amplitude of one. This is done to simplify the target recognition process, as discussed below.

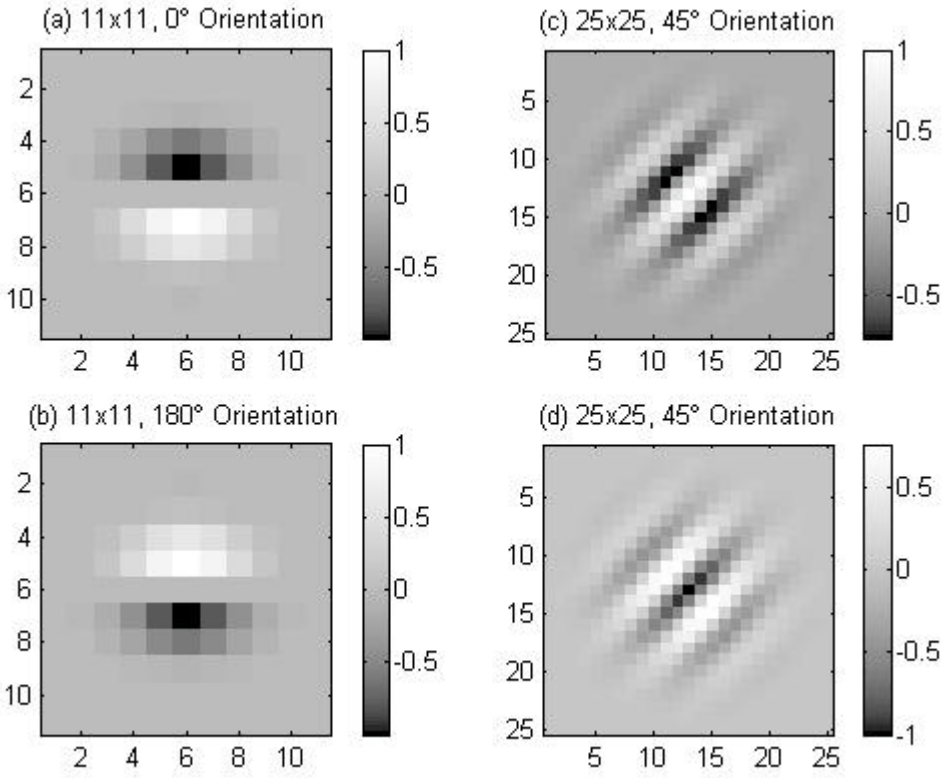


Figure 3-4. Examples of Gabor kernels associated with orientation layers. (a) Edge Selective Kernel, spatial frequency = 15 cycles per 100 pixels, 90º phase, s=1.5, (b) Edge Selective Kernel, spatial frequency = 15 cycles per 100 pixels, 90º phase, s=1.5, (c) Bright Line Selective Kernel, spatial frequency = 20 cycles per 100 pixels, 0º phase, s=4.0, (d) Dark Line Selective Kernel spatial frequency = 20 cycles per 100 pixels, 0º phase, s=4.0. The kernel size is given above the respective figure.

3.1.3 Layer/Kernel Bias

The algorithm used by ROCIT is loosely based on the constructive hierarchical model of *Hubel and Wiesel* [1977]. In this paradigm, sets of adjacent retinal ganglion cells oriented along a particular trajectory project to a simple cell of layer 4 in the visual cortex, causing it to be selective to oriented edges[8], and higher level features are constructed from lower ones as information is transmitted along the ventral stream. Figure 3-5 shows how three spatially adjacent on- or off-kernels can be summed to produce a kernel that would be selective for a

[8] For an alternative and modern interpretation of this simplistic formulation, see *Martin* [2002]. He reviews the important role played by inhibitory interneurons in producing the selectivity of simple and complex cells in layers 3 and 4, respectively. As well, he emphasizes the need to understand the processing role of lateral and feedback connections as opposed to a purely feed forward model.

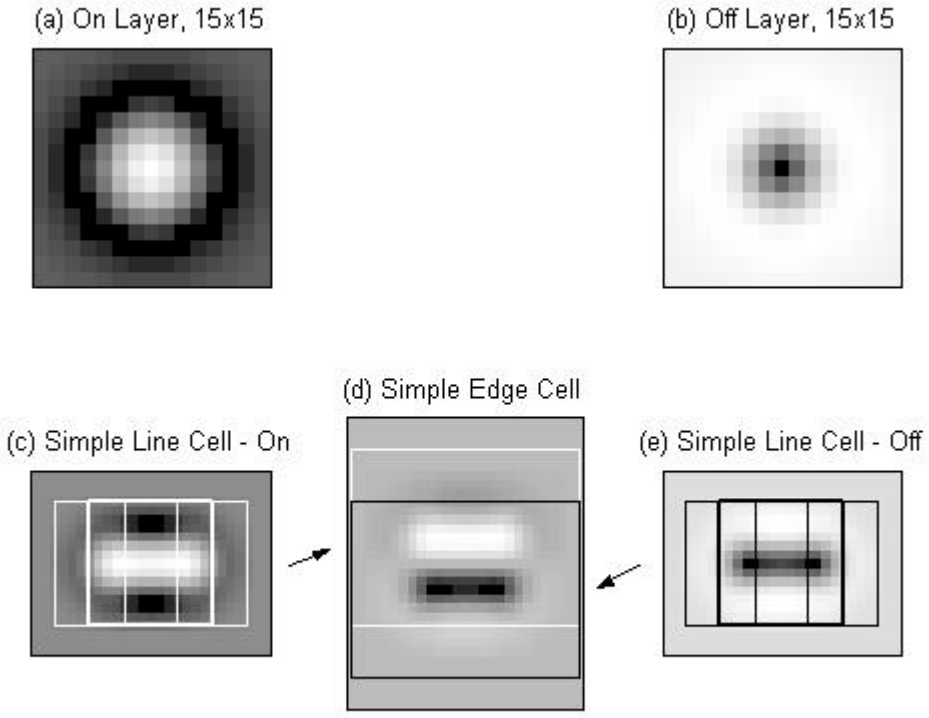


Figure 3-5. Construction of simple cells through a linear combination of on and off kernels. A linear arrangement of on kernels (a) can produce a kernel that will be selective to bright line segments (c), while a linear arrangement of off kernels (b) produces a kernel selective to dark line segments (e). These may be combined to produce an edge selective kernel (d).

Layer Selectivity	Bias	Layer Selectivity	Bias
On	+1	-	
Off	-1	-	
0º Line	+1	0º Edge	+1
45º Line	+1	45º Edge	+1
90º Line	+1	90º Edge	-1
135º Line	+1	135º Edge	-1
180º Line	-1	180º Edge	-1
225º Line	-1	225º Edge	-1
270º Line	-1	270º Edge	+1
315º Line	-1	315º Edge	+1

Table 2-1. Bias values for on/off layers and for line- and edge-selective orientation layers.

horizontal line segment (Figure 3-5 (c), (e)). A combination of these can produce an edge selective cell (Figure 3-5 (d)).

The challenge is to develop a synaptic weight pattern that can take ranked spikes coming from on/off layers and use them as "evidence" matching the selectivity of a given orientation layer. ROCIT adopts the method of SpikeNET, which uses the synaptic weights of the efferent layer. This is counterintuitive and requires the introduction of a "bias" in the synaptic connection so that the nature of the afferent layer (on vs. off) can be taken into account. Spikes from the on layer are evidence of a bright line segment whereas those from off layers are evidence of a dark one. Since on and off layers use kernels that differ by a factor of –1, a "bias" value of –1 is introduced for the off layer. With this formulation, the spike from an off layer neuron causes a linear inhibition in an orientation layer selective to bright line segments while a on layer spike causes excitation, and vice versa.

The same reasoning leads to the use of a bias associated with orientation layers. Line selective orientation layers sharing the same orientation but opposite amplitudes have opposite biases. Edge selective orientation layers with 180º different orientations also have opposite biases (*e.g.*, Figure 3-4 (a) and (b)). A summary of the bias values used by ROCIT are given in Table 2-1. There is no mention of bias in the SpikeNET publications [*Delorme et al.*, 1999; *Delorme and Thorpe*, 2001a & 2001b; *Thorpe et al.*, 2001], however, in a GNU public license release of an early version SpikeNET[9] there are kernels of opposite sign used for synaptic connections between the on/off layers and any single orientation layer. This indicates that the bias has been built into the selectivity kernel. In ROCIT the bias value is used in two critical algorithm components: updating target kernels and voltage updates. This is discussed in more detail in subsequent sections.

3.2 Algorithm Components

The rank order coding algorithm is straightforward and consists of a few simple components shown in Figure 3-6. The four primary components are (1) converting the image into ranked spikes in the on/off layers, (2) updating the target kernel (Training Mode only), (3) updating voltage values, and (4) updating sensitivity values. The following sections document the details of these portions of the algorithm.

3.2.1 On/Off Layer Preprocess

The image is converted into ranked spikes by convolving it with a given on/off kernel, sorting the resulting values, and assigning them to bins of equivalent rank. Figure 3-7 shows an example of the convolution process. As can be seen, edges are enhanced as are small areas of brightness contrast such as eye reflections (Figure 3-7 (b)), and nostrils (Figure 3-7 (c)).

The values of the convolved images are sorted and then divided into a fixed number of bins. The number of bins is referred to as the "Maximum Time Steps", although time is only

[9] There is a GNU general public license, version 2 release of SpikeNET available at Arnoud Delorme's web page at: http://sccn.ucsd.edu/~arno/spikenet/index.html, Accessed April, 2004.

(a) Training Mode Algorithm

```
Convert the Image into Ranked Spikes in the On/Off Layers
while (Spikes Occur || Stopping Criteria is Reached)
{
      Gather Spike Lists from Afferent Layers
      Process Spikes in Efferent Layers:
            if Efferent Layer == Target Layer
                  Update Target Kernel
                  Update Sensitivities
            else
                  Update Voltages
                  Check Voltage Thresholds for New Spikes
                  Update Sensitivities
            end
}
```

(b) Recognition Mode Algorithm

```
Convert the Image into Ranked Spikes in the On/Off Layers
while (Spikes Occur || Stopping Criteria is Reached)
{
      Gather Spike Lists from Afferent Layers
      Process Spikes in Efferent Layers:
            Update Voltages
            Check Voltage Thresholds for New Spikes
            Update Sensitivities
}
```

Figure 3-6. Essential components of the rank order coding algorithm. (a) Training Mode, (b) Recognition Mode.

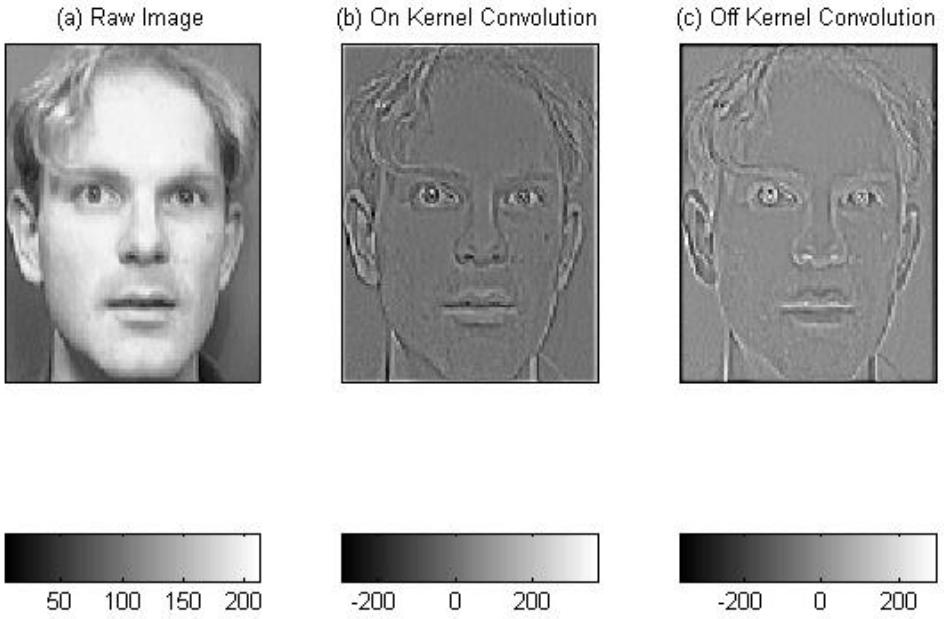


Figure 3-7. Example convolutions used to generate ranked spikes for the on/off layers. (a) Raw Image, (b) On Layer Convolution, and (c) Off Layer Convolution. Note: The on/off kernels used were those of Figure 3-3 (a) and (b), respectively.

relative in the algorithm. This value is typically set to 500. If, however, pixels have the same value, then they are forced to share the same bin. This is the same algorithm as used by SpikeNET [*Delorme et al*., 1999] with the exception of the bin sharing. Failure to put similar values into the same bin would result in the spikes having different impacts on their efferent layers. This is because an efferent neuron may have undergone desensitization between the arrival of the two spikes. Another difference is that SpikeNET only allows either the on or off neuron at a given layer location to fire [*Delorme and Thorpe*, 2001a]. This constraint

was omitted from ROCIT because there was no physiological basis for it. It should also be noted that neurons along a border of the same width as the on/off kernel are excluded from the spike ranking process since the convolution values there suffer from edge effects. These edge effects can be seen in Figure 3-7 (b) and (c).

3.2.2 Voltage Updates

A neuron's voltage is modified if a spike occurs in any afferent layer that the neuron has a synaptic connection to. Figure 3-8 depicts the geometry associated with voltage updates. The voltage change at the efferent neuron $\Delta V_{eff}(i, j)$, is proportional to the selectivity kernel of the efferent layer and is given by:

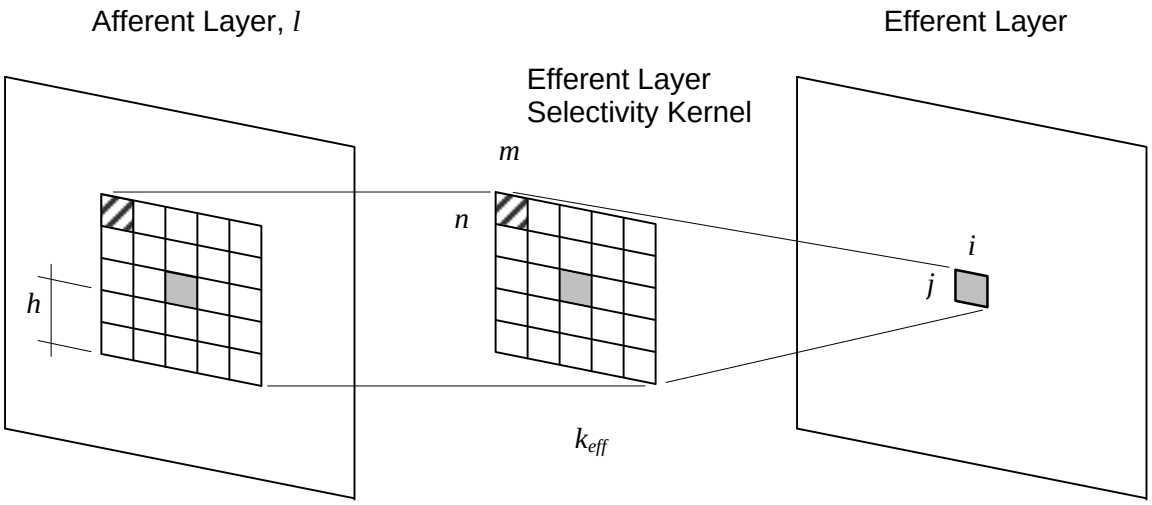


Figure 3-8. Synaptic geometry associated with voltage updates. The afferent layer has a spiking neuron at the location with cross-hatching. Note that *h* is the half width of the kernel minus one, assuming an odd sized, square kernel.

$$\Delta V_{eff}(i,j) = \sum_{l=1}^{\substack{afferent\\layers}} \sum_{m=i-h}^{m=i+h} \sum_{n=j-h}^{n=j+h} d_{aff}^{l}(m,n) \cdot b_{aff}^{l} \cdot k_{eff}(m-i+h+1, n-j+h+1) \cdot S_{eff}(i,j) \quad (6)$$

Indices (i, j) indicate a position within a neuron layer while (m,n) indicate a position within the efferent layer selectivity kernel. $d_{aff}^{l}(m,n)$ is a Dirac delta function defined over the relevant neurons of afferent layer *l* such that

$$d_{aff}^{l}(m,n) = \begin{cases} 1 & if\ a\ spike\ occurs\ at\ neuron(m,n)\ of\ afferent\ layer\ l \\ 0 & otherwise \end{cases} \quad (7)$$

b_{aff}^{l} is the bias of the afferent layer, k_{eff} is the selectivity kernel of the efferent layer, and $S_{eff}(m,n)$ is the sensitivity of the efferent neuron at the location of the afferent spike. Note that this formulation assumes that the kernel has an odd dimension and is and square, although these assumptions are relaxed in ROCIT for the target layers.

While this formulation appears complex, the results are straightforward. Figure 3-9 shows examples of how afferent spikes coming from on and off layers affect the voltage of an orientation layer selective to a vertical edge. Because of the "symmetric" synaptic connections used by ROCIT (see Figure 3-2), an afferent spike causes voltage changes in a

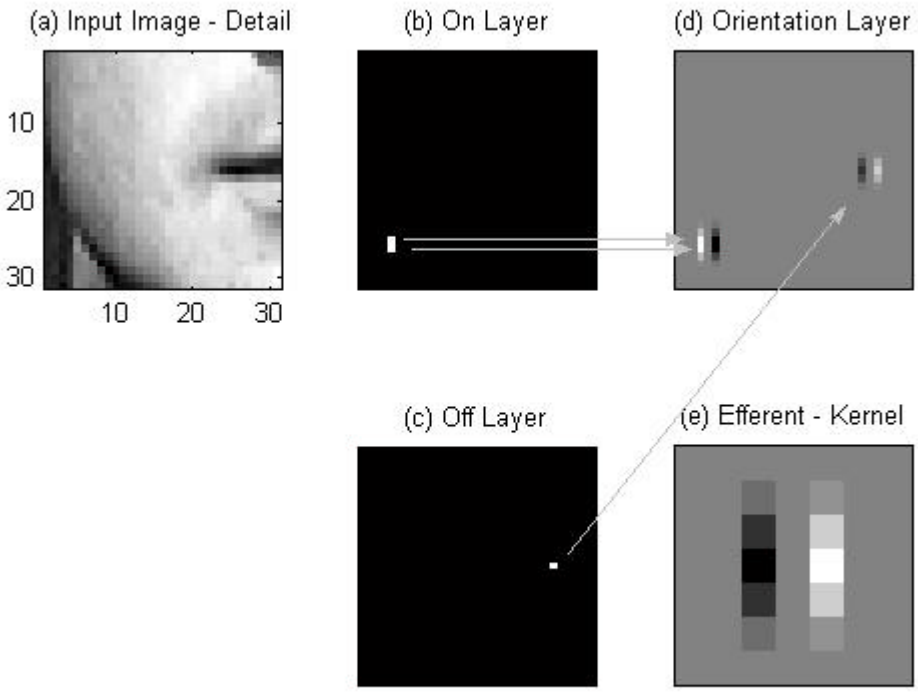


Figure 3-9. Details of how spike propagation from on/off layers affect voltage values in an orientation layer. (a) Detail of an input image, locations of afferent spikes from the on (a) and off (b) layers, (d) corresponding voltage changes in the orientation layer, and (e) the selectivity kernel of the orientation layer.

kernel shaped area in each of its efferent layers. The voltage update appears as a selectivity kernel that has been flipped up and down as well as left to right.

Bias values act to reverse the resulting pattern. As seen in 3-9 (d), spikes from different afferent layers produce reversed voltage updates since they have opposite bias values (`+1` for the on layer and `-1` for the off layer). Adjacent spikes act to reinforce each other one another's voltage effects. In this example, the two spikes from the on layer are aligned vertically, so the voltage updates overlap, causing reinforcement in both the positive and negative regions.

Voltage values are compared to a threshold. Those neurons whose voltage exceeds the threshold fire a spike and are reset to zero. The spike is placed in the next rank bin for propagation to its layer efferent layers.

Figure 3-10 shows an orientation layer after 40 of 500 rank bins have fired from the on/off layers. The orientation layer is sensitive to horizontal edges with dark shading above light (Figure 3-10 (d)). As can be seen in the voltage pattern, the on/off spikes have originated

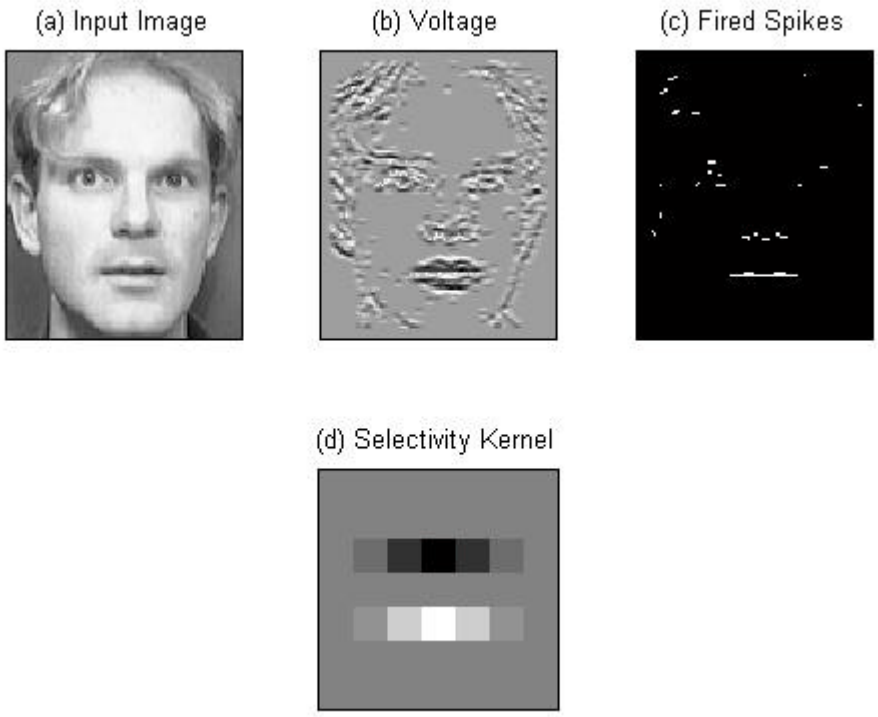


Figure 3-10. Voltage development in a horizontal orientation layer. (a) Input image, (b) voltage after 40 of 500 rank bins have been propagated from the on/off layers, (c) neurons firing in the orientation layer, and (d) selectivity kernel of the orientation layer.

from the dominant facial features (eyes, nose, ears, *etc.*), the hair, and the chin. The mouth and nostrils are seen to have dark-above-light horizontal edges, which has caused the corresponding orientation layer neurons to fire. Other, less distinct areas of horizontal edges are seen near the eyes and in the hair.

3.2.3 Target Kernel Development (Training Mode Only)

A training algorithm, distinct from the recognition algorithm, is used to develop a custom kernel for a specified target, and only a single target can be trained at a time. During training a single target layer is added to the network. This layer is initialized with a kernel matrix of zero values. Only sensitivity is modeled for the target layer since the training algorithm is dedicated toward constructing the target kernel rather than developing a voltage pattern for target detection. Figure 3-11 depicts the geometry associated with custom kernel updates.

The kernel is modified according to

$$\Delta k_t(m,n) = \sum_{l=1}^{\substack{afferent\\layers}} \sum_{m=i-h}^{m=i+h} \sum_{n=j-h}^{n=j+h} d_{aff}^{\,l}(i,j) \cdot k_{aff}^{\,l}(m-i+h+1, n-j+h+1) \cdot S_t(m,n) \qquad (8)$$

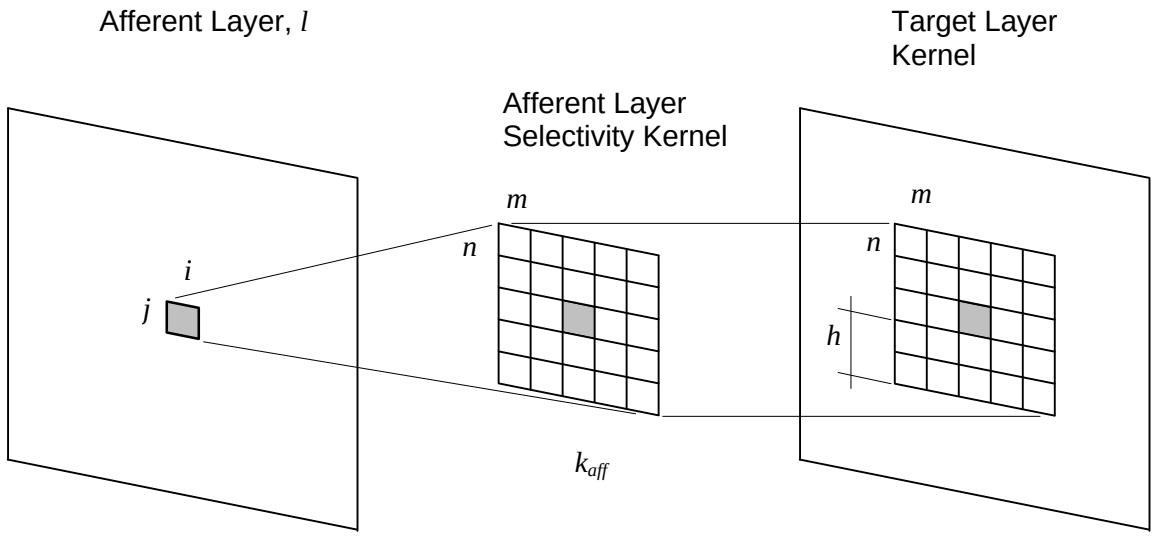


Figure 3-11. Synaptic geometry associated with custom target kernel training. The afferent layer has a spiking neuron at the location of the shaded square. This results in target kernel modifications in an area corresponding to the afferent layer's selectivity kernel. Note that *h* is the half width of the kernel minus one, assuming an odd sized, square kernel.

where $\Delta k_t(m,n)$ is the change in the target kernel at position (m,n), $d_{aff}^{l}(i,j)$ is a Dirac delta function over the afferent layer *l* as defined by

$$d_{aff}^{l}(i,j)=\begin{cases}1 & \textit{if a spike occurs at neuron}(i,j)\textit{ of the afferent layer } l\\ 0 & \textit{otherwise}\end{cases} \tag{9}$$

k_{eff} is the selectivity kernel of the efferent layer, and $S_t(m,n)$ is the sensitivity of the target neuron at the location of the kernel update. Note that this formulation assumes that the kernel is odd and square. ROCIT uses odd, square kernels in the orientation layers and these layers are typically the only afferent layers for a target layer.

This formulation has the effect of adding a weighted afferent kernel to the target kernel at a point centered at the locus of the afferent spike. The weight consists of the product of the afferent layer bias and the sub-matrix of sensitivity values of the target layer that aligns with the afferent layer's selectivity kernel. Figure 3-12 shows the development of a custom target kernel at a series of time steps. Each step corresponds to propagating all spikes in a particular rank bin. The earliest spikes arrive from the area around the eyes. These spikes result in the highest weights in the custom kernel. Later spikes highlight the left ear, neck, and mouth.

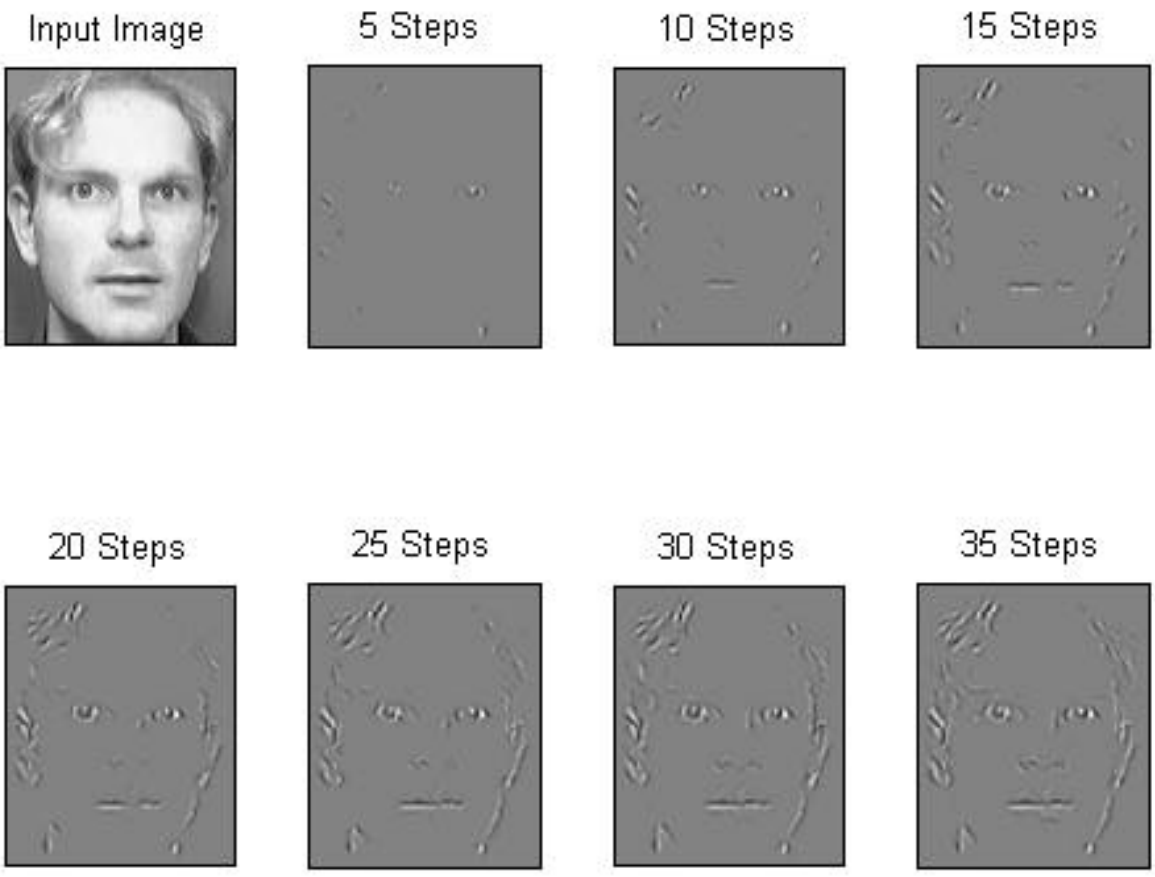


Figure 3-12. Custom target kernel development. The target kernel resembles its target image, but areas lacking lines or edges are de-emphasized. Areas having the earliest spike arrivals have the highest weight development.

When this kernel is used for recognition, spikes arriving from locations of high weight values will cause the greatest voltage increases. This behavior is enhanced by the use of desensitization produced by mechanism that mimics shunting inhibition.

3.2.4 Sensitivity Updates

Shunting inhibition is the key mechanism invoked by *Gautrais and Thorpe* [1998] both to encode and decode spike sequences. Conceptually, shunting inhibition is the desensitizing of neurons after they receive an excitatory stimulus. If spikes do not arrive in order from highest to lowest synaptic weight, then the voltage increase at the efferent neuron will not be maximal because of the impact of shunting inhibition. Figure 3-13 shows how this can occur via a lateral inhibitory connection.

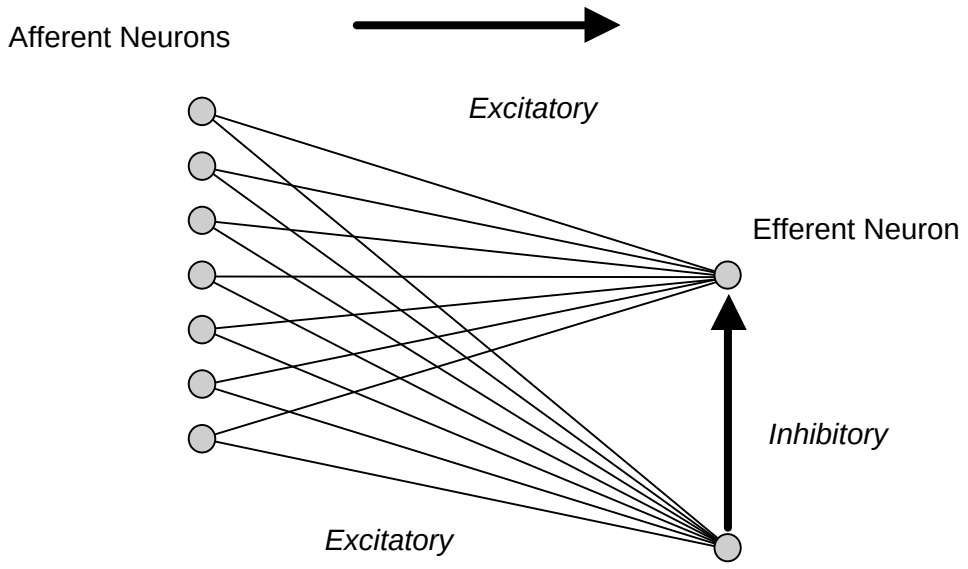


Figure 3-13. Synaptic circuit implicated in shunting inhibition. Shunting inhibition causes a neuron to be desensitized after it receives an excitatory stimulus.

ROCIT's algorithm applies desensitization following the voltage updates and threshold checking. There are two distinct modes of desensitization: global and local. Global desensitization is when an entire layer of neurons is desensitized whenever any neuron in the layer receives an afferent spike. Local desensitization is when only those neurons whose voltages were modified undergo desensitization.

Global desensitization is used to develop new target kernels and was used in the simulation shown in Figure 3-12. This form uses a homogeneous scalar value of sensitivity for an entire target layer that decreases with spike rank:

$$S_t(i,j) \equiv S_t = a^{\,rank} \tag{10}$$

subject to $0 < a \leq 1$. Here a is the initial sensitivity and *rank* is the rank of a given spike rank bin. Figure 3-14 shows how the sensitivity decays with rank as a function of the initial sensitivity, a. Global desensitization will bias the custom target kernel by spike rank, thus encoding spike order as per Equation (8).

Local desensitization is only used during recognition mode, and only if the targets exist in a cluttered scene. If global desensitization were used, and if the spikes associated with a target were located late in the ranked bins, then the target layer would be desensitized by the time its spikes arrived and the target would be missed. Local desensitization overcomes this

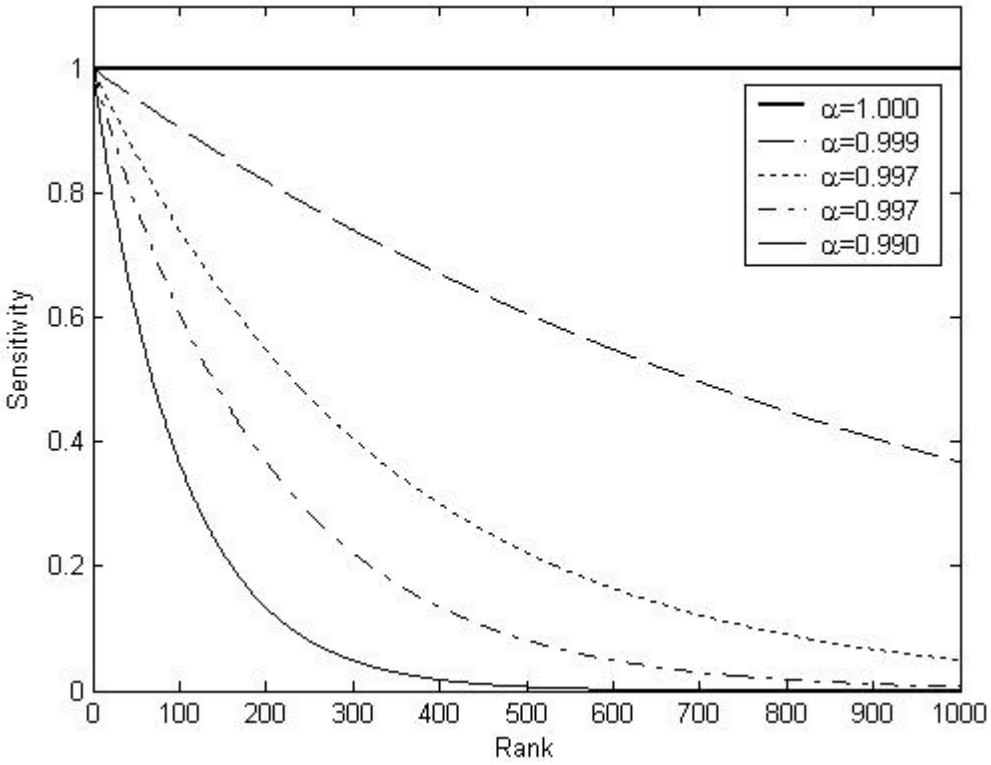


Figure 3-14. Global desensitization as a function of rank and initial sensitivity, a.

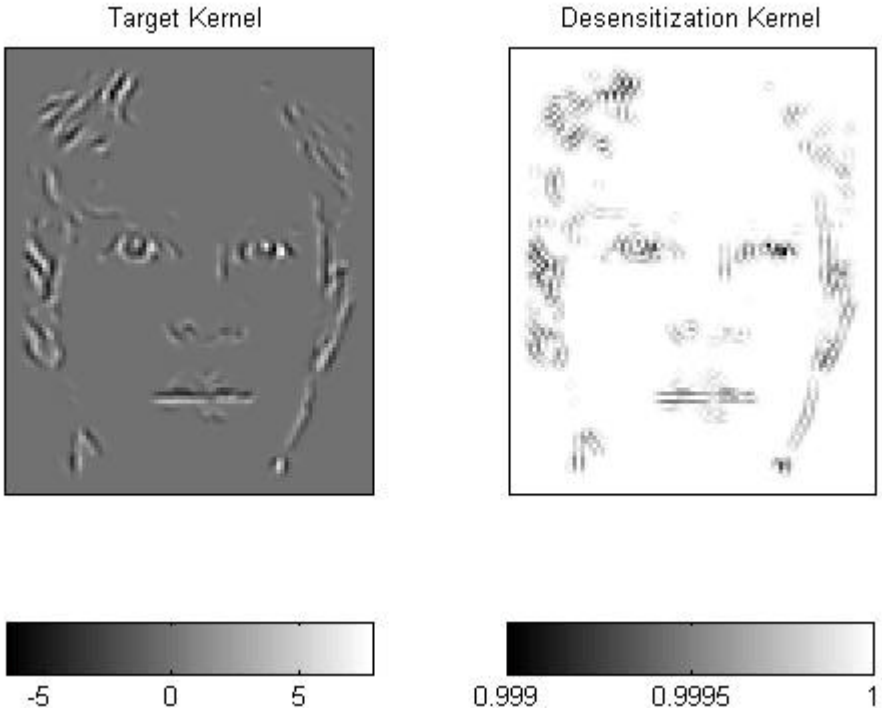


Figure 3-15. An example local desensitization kernel and its corresponding target kernel.

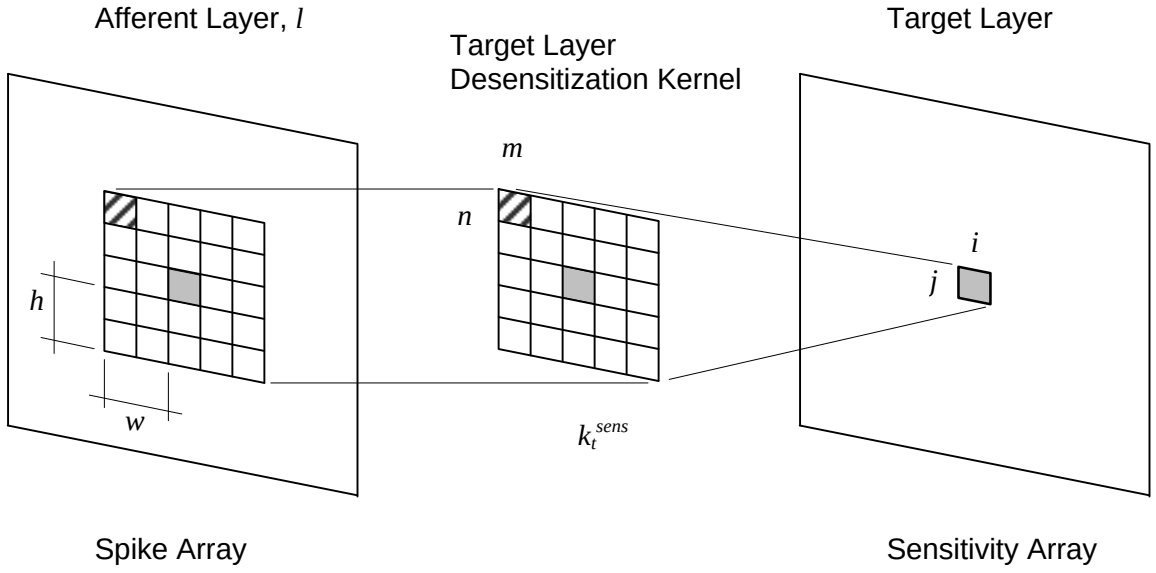


Figure 3-16. Synaptic geometry associated with sensitivity updates. The afferent layer has a spiking neuron at the location with cross-hatching. Note that *h* is the half height of the kernel minus one, and *w* is the half width of the kernel minus one, assuming an odd sized.

problem. It uses a "desensitization kernel" that is developed from the custom target kernel by:

$$k_t^{sens}(m,n) = \frac{(a-1)}{\max(abs(k_t))} k_t(m,n) + 1 \tag{11}$$

again, where the initial sensitivity is $0 < a \le 1$. This formulation assigns a value of a to the peaks (max or min) of the target kernel and decreases to a background value of one as the absolute value of the target kernel decreases. Figure 3-15 shows a typical desensitization kernel.

The sensitivity update for local desensitization is similar to voltage updates. Figure 3-16 depicts the geometry associated with sensitivity updates. The sensitivity change at the target layer neuron is proportional to its desensitization kernel. It is given by:

$$S_t(i,j) = \sum_{l=1}^{\substack{afferent \\ layers}} \sum_{m=i-w}^{m=i+w} \sum_{n=j-h}^{n=j+h} d_{aff}^{l}(m,n) \cdot k_t^{sens}(m-i+w+1, n-j+h+1) \cdot S_t(i,j) \tag{12}$$

Indices (i,j) indicate a position within a neuron layer while (m,n) indicate a position within the efferent layer selectivity kernel. As before, $d_{aff}^{l}(m,n)$ is a Dirac delta function defined over the relevant neurons of afferent layer *l* such that

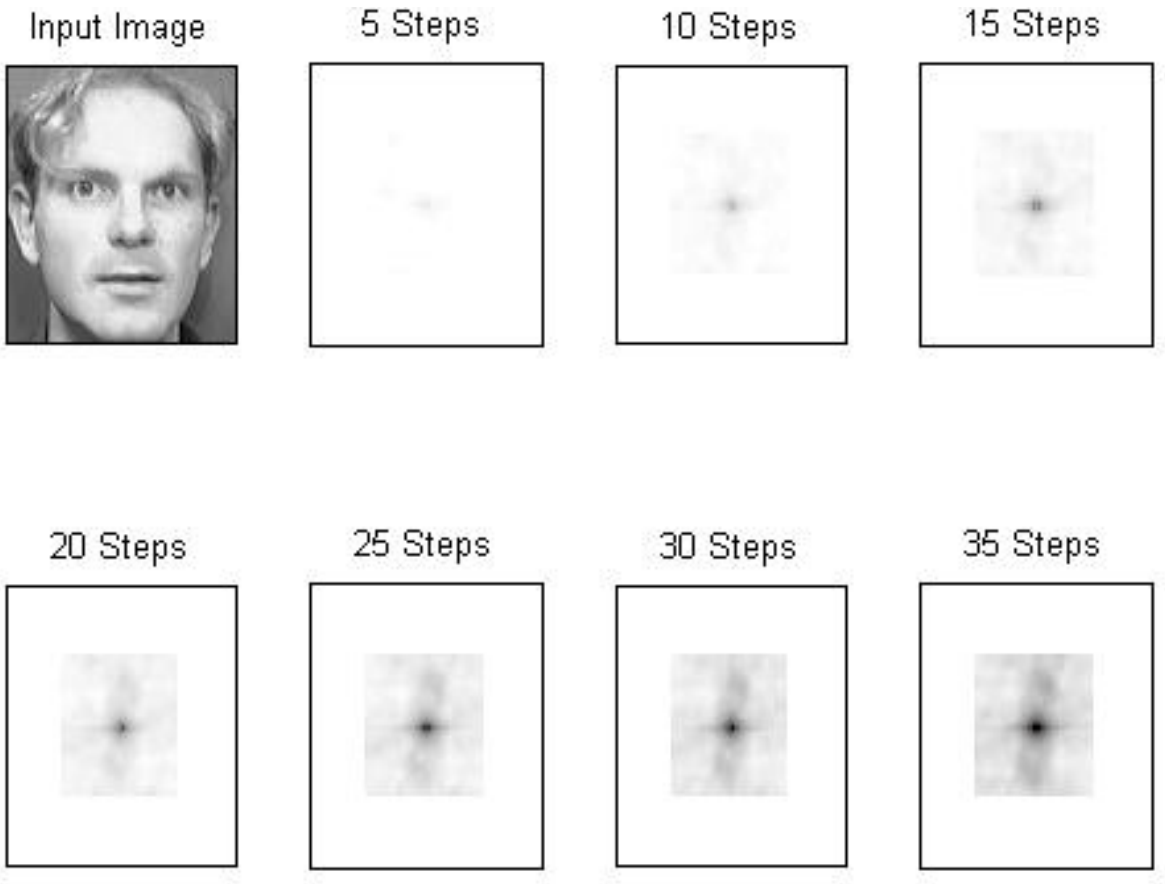


Figure 3-17. Target layer desensitization development under local sensitivity updates. Desensitization becomes proportional to the weight associated with a voltage update. This focuses desensitization to the neuron corresponding to the center of the target.

$$d_{aff}^{l}(m,n)=\begin{cases}1 & \textit{if a spike occurs at neuron}(m,n)\textit{ of afferent layer } l\\ 0 & \textit{otherwise}\end{cases} \tag{13}$$

and k_t^{sens} is the desensitization kernel for the target layer. Note that this formulation assumes that the kernel is odd and rectangular, although these assumptions may be easily relaxed.

Figure 3-17 shows the development of neuronal desensitization in a target layer using local desensitization. As can be seen, the region of desensitization is localized to the center of the target. One can also see that ROCIT does not update the sensitivity (or voltage) of a target layer in a border region equal to one quarter of the target width and height. This is solely for the purpose of speed, and risks missing targets near the edge of an image.

ROCIT allows the initial sensitivity of all layers to be set independently. This enables the user to choose which layers should undergo desensitization. It is not presently clear how desensitization of orientation layers impacts recognition, so this flexibility was added to enable future studies.

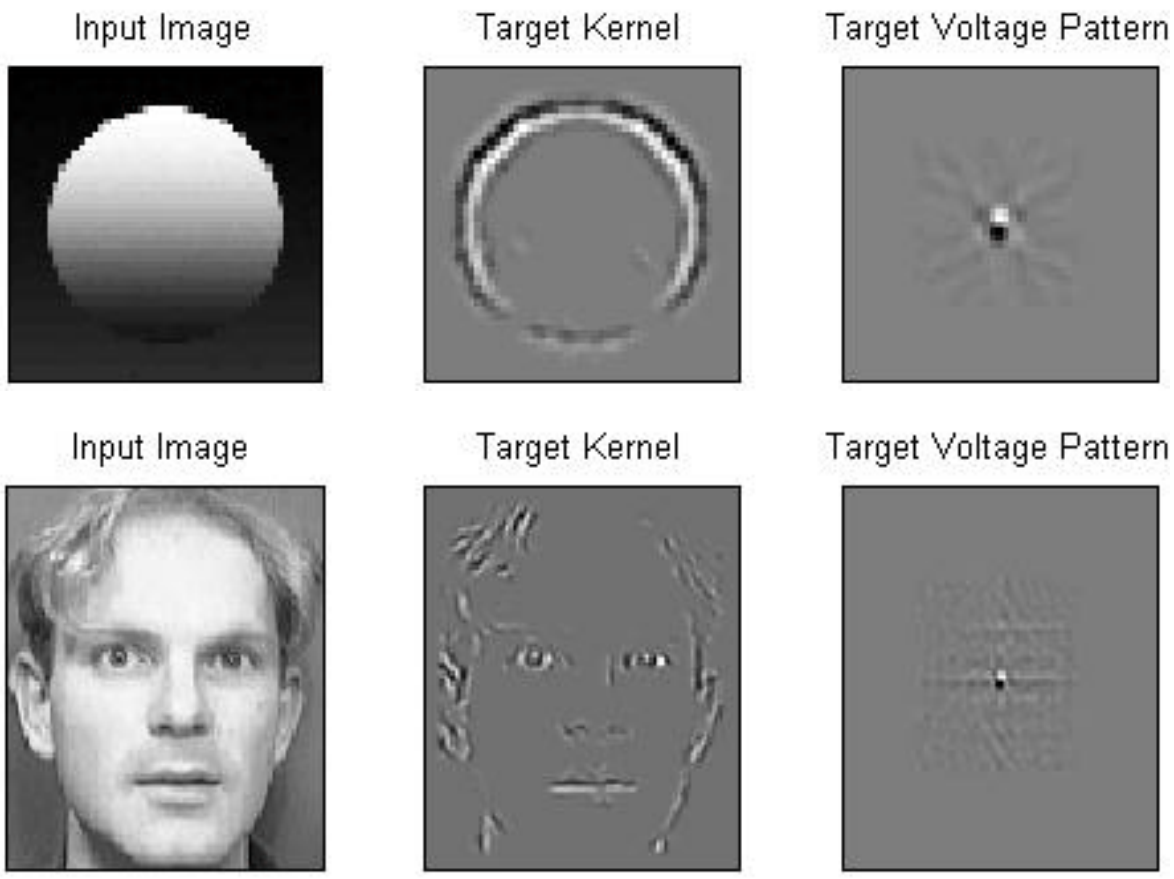


Figure 3-18. Examples of training images, their kernels, and the voltage pattern they produce when run through recognition mode. Note the pairing of a positive peak and negative trough at the center of the target.

3.2.5 Kernel Normalization and Voltage Thresholds

Different targets have different sizes and shapes, resulting in different target kernels. Each target kernel creates a unique voltage pattern when the training image is run through recognition mode. The voltage pattern contains a peak at the center of the voltage layer with a trough just below. Figure 3-18 shows two examples of this voltage pattern.

The paired peak-trough results from the asymmetry of the bias values used for the edge-selective orientation layers as shown in Figure 3-19. There is no way of eliminating this asymmetry with eight layers at 45º increments, so the bias values have been chosen to create a voltage pattern that is nearly symmetric about the vertical for objects having lateral symmetry.

Ideally, a single voltage threshold should be applied to all target layers, otherwise, the neurons in the target layers are not homogeneous. To accomplish this, ROCIT first performs recognition of a training image using its raw target kernel. Then, the kernel is normalized by the maximum voltage that develops when it is used in recognition mode. In this way, target

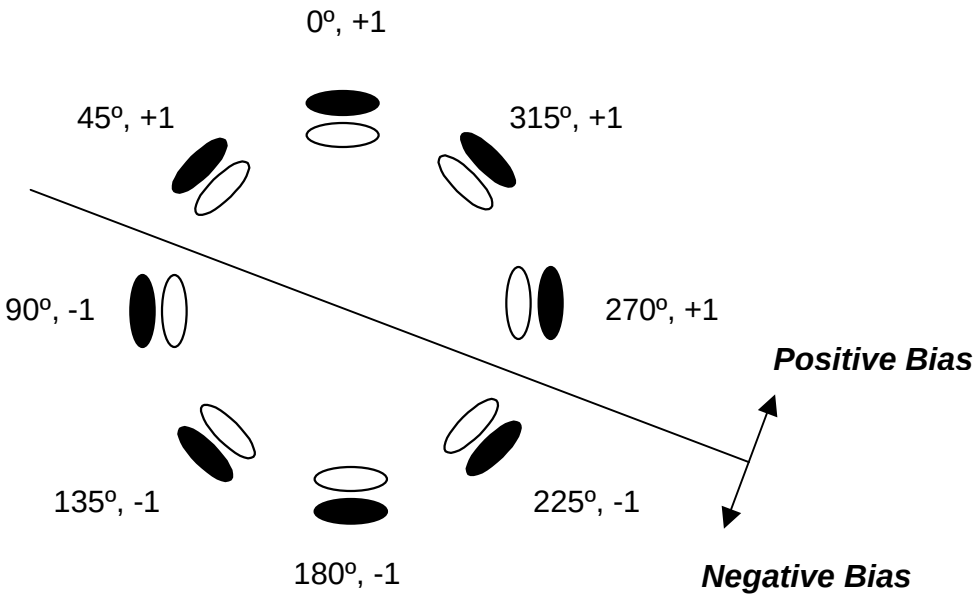


Figure 3-19. Bias values associated with edge-selective orientation kernels. Note the asymmetry of the bias pattern. This is responsible for the paired peak-trough patterns that develop in target layer voltages. This asymmetry is unavoidable with eight orientation layers at 45º increments.

kernels can be counted on to produce a maximum voltage of 1.0 when their respective training image is presented. Lower voltage values should develop whenever the image is degraded with noise or occlusion.

As noted previously, on/off kernels are normalized to have a maximum absolute peak of +1 as are orientation layer kernels. Voltage increases in the orientation layers depend on the constructive interference of their respective kernels. ROCIT typically uses a voltage threshold of 2.5 for orientation layers implying at least three spatially-coherent on/off afferent spikes to activate a given orientation layer neuron.

3.2.6 Recognition Algorithm

The recognition algorithm is identical to the training algorithm for the on/off and orientation layers. However, a target layer is instantiated for each target to be sought. During recognition, the voltage, voltage threshold, and sensitivity of each target layer is fully modeled. There are several ways of detecting a target: when a voltage exceeds a prescribed threshold, when a voltage pattern matches the one known to be caused by the target, or associating local voltage maxima with target detections. ROCIT allows the user to choose between these three algorithms.

Using voltage maxima is straight-forward, but risks mislocating the target by a pixel or two since the true target center lies between the peak and trough of the voltage pattern. This can be seen in the upper right image in Figure 3-18 and stems from the bias values associated

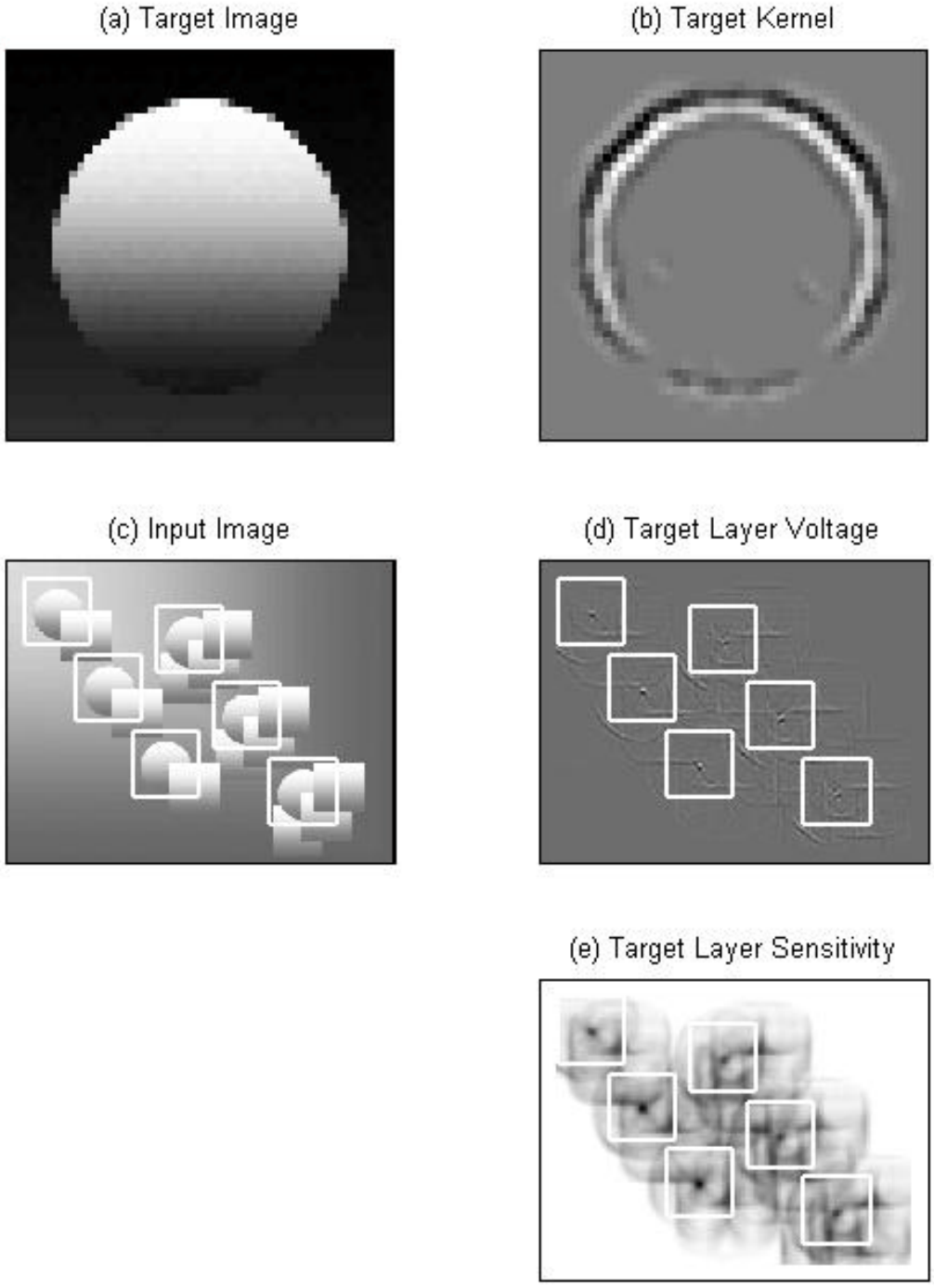


Figure 3-20. Voltage development, sensitivity development, and target detections for a diagnostic image containing multiple, partially occluded, shaded targets (a) Raw target used for training, (b) Target kernel, (c) Input image, (d) Voltage development, and (e) Sensitivity development. White squares indicate where the target was detected and located by the recognition algorithm.

with the orientation layers. As discussed in Section 3.1.3, there are bias values, $bias \in \{-1,+1\}$, associated with orientation layer kernels that are used during target layer voltage updates. The asymmetry of these values (See Figure 3-18) causes the peak/trough pair and the way the biases are arranged determines whether the pair with be vertical, horizontal, or some other orientation.

This pattern can be leveraged to improve recognition. In this case the known target pattern is first cross-correlated with the target layer voltage. The resultant values may then be thresholded. The result will place a target center between the peak/trough pair.

With either method, ROCIT iteratively finds the maximum peak, masks out an area of one fourth the target kernel size centered on the peak, and searches for the next. This continues until either a fixed number of targets is found or until a minimum peak magnitude is encountered. Figure 3-20 shows a typical example including voltage development, sensitivity development, and target locations in a cluttered scene of shaded circles and squares. The white squares mark the outlines of where the target was found in the input image. At this point all six targets are correctly detected and located. Distinct voltage peaks indicate the center of the targets (Figure 3-20(d)). Desensitization has also been focused on these locations (Figure 3-20(e)).

This page intentionally left blank.

4. Implementation Overview

ROCIT is implemented in MATLAB® 6.5. There is an accompanying graphical user interface (GUI), also in MATLAB® (See Figure 3-21). MATLAB® is an excellent algorithm prototyping framework, and has enabled us to easily explore modifications to and extensions of the rank order coding algorithm. A faster C++ implementation (SpikeWave) with a JAVA GUI is also under development.

Both ROCIT and SpikeWave use three types of input files: a network specification file, an image file, and a custom target file. All are written in XML to provide easy parsing, clear parameter labels, and extensibility while providing a common format for both implementations. An example network specification file is given in Appendix C. Image files formats include JPEG, GIF, TIFF, PGM, BMP, or text data embedded in a ROCIT XML file. When custom targets are trained, their kernels are saved in a target file that includes all the network specifications used in its development. The training image and expected voltage pattern are included in the custom target file. By including network metadata in the target files, the current network settings may be compared to the target's to ensure that they are consistent. An example target file is listed in Appendix D.

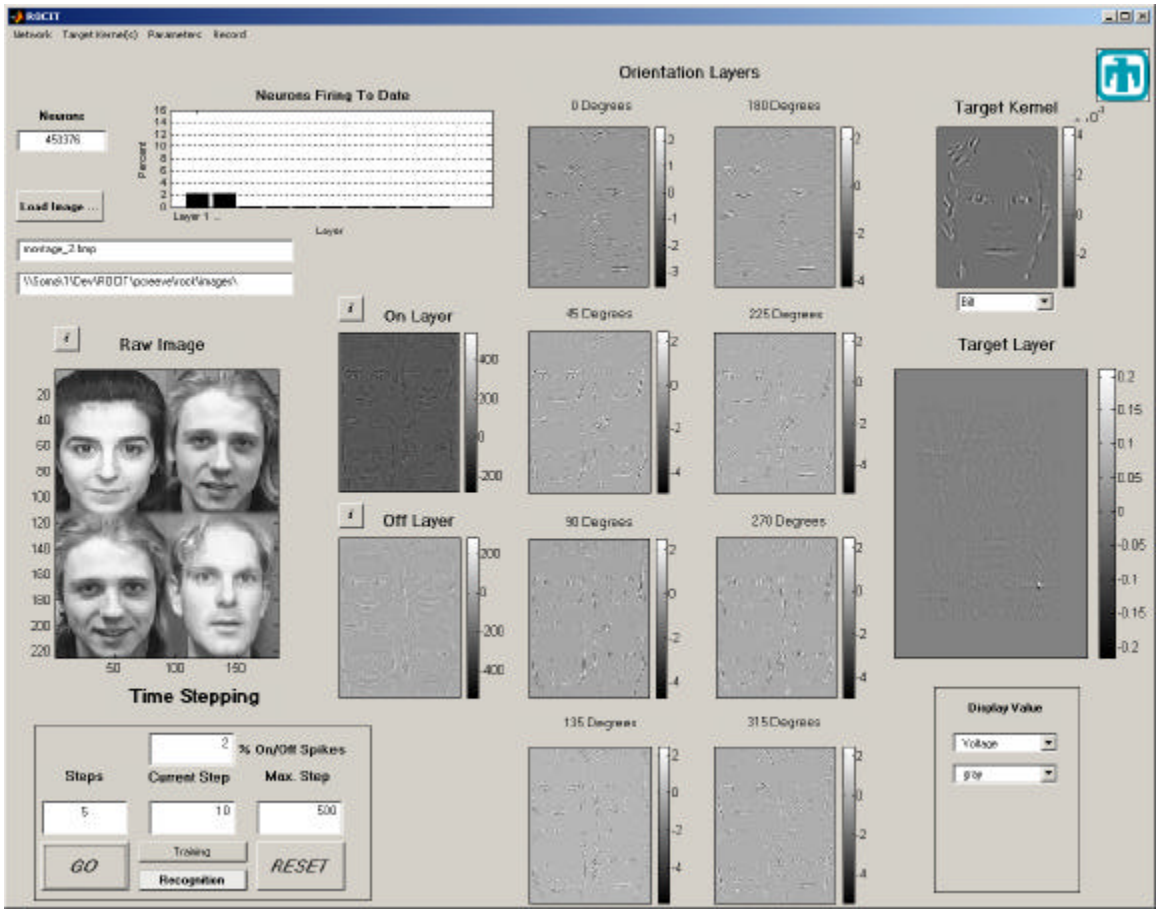


Figure 4-1. Screenshot of the ROCIT GUI implemented in MATLAB.

This page intentionally left blank.

5. Initial Benchmarking

To date, the performance of ROC schemes with respect to object recognition have been anecdotal (*e.g.*, *Delorme and Thorpe* [2001a], *Van Rullen et al*. [1998], *Muresan* [2002a], *Muresan* [2003]). We explored the issue of recognition performance using a benchmark problem developed for the object recognition domain of biometrics. We are not interested in biometrics, *per se*, but the benchmark chosen explores recognition capabilities across a wide set of similar objects (faces) with variable lighting and moderate plasticity (facial expressions), which is a subset of the general problem of object recognition in cluttered, uncontrolled environments. SpikeWave, the C++ implementation of the baseline ROCIT algorithm was used for the benchmark calculations.

5.1 FERET Database

The Face Recognition Technology[10] (FERET) program was initiated to develop automatic systems that could be used to assist law enforcement, security, and intelligence personnel perform face recognition and verification tasks [*Phillips et al*., 1996]. The program ran from 1993 through 1997 with sponsorship from the U.S. Department of Defense Counter Drug Technology Development Program. During this five year span, a large database of facial images and a number of testing protocols were developed to gauge the performance of face recognition algorithms. The tests have been administered on various face recognition algorithms from laboratories and universities across the United States, and the results for each algorithm are well published.

5.2 FERET Verification Testing Protocol

The FERET Verification Testing Protocol [*Rizvi et al*., 1998] was one of the tests developed and administered by the FERET program. A verification system confirms the claimed identity of a face presented to it. The FERET test uses a database of images to test recognition performance with a number of different constraints. In the FERET Verification Testing Protocol, the image of the person claiming an identity is termed a *probe image*. The set of known individuals the system searches through to confirm the identity of the probe is known as the *gallery image set*. Various image sets exist within the FERET database, and each tests algorithm performance under different circumstances.

We chose the **fafc** subset of the image dataset to benchmark because it tests algorithm performance with variable lighting conditions and moderate pose and expression variations. Example images from this benchmark may be seen in Figure 5-1. The images are eight bit grayscale stored in Portable Network Graphics (PNG) format, and are 384 pixels high by 256 pixels wide. The **fafc** test uses 1196 gallery images from the **fa** subset of the FERET database, and 194 probe images from the **fc** subset. The **fc** probe images were taken on the same day as the **fa** images, however the **fc** images were taken with a different camera under different lighting conditions. In the test, the system is first trained to recognize all 1196

[10] The Facial Recognition Technology (FERET) Database, NIST, http://www.itl.nist.gov/iad/humanid/feret/feret_master.html, Accessed April, 2004.

Figure 5-1. Example images of individuals taken from the FERET fafc benchmark. The images on top are used for training and the bottom images are presented to the system for identification.

gallery images. Then, each probe is compared against each image in the known set of gallery images. Each comparison produces a match score. If this match score is above a threshold, then recognition system returns true, otherwise the match is false. This output can be classified in one of four categories based on the actual identity of the individual. These categories are shown in Table 5-1.

The results from the FERET Verification Test can be compared by a receiver operating characteristic (roc) curve (not to be confused with Rank Order Coding). The *y*-axis of a roc curve represents the verification probability, denoted by P_V as defined by Equation (14). The *x*-axis represents the false alarm rate, denoted by P_F as defined by Equation (15):

System Output	Actual Identity	Output Classification
$p(i) = g(i)$	$p(i) = g(i)$	True Positive (TP)
$p(i) \neq g(i)$	$p(i) = g(i)$	False Negative (FN)
$p(i) = g(i)$	$p(i) \neq g(i)$	False Positive (FP)
$p(i) \neq g(i)$	$p(i) \neq g(i)$	True Negative (TN)

Table 5-1. The four different possible outcomes as a function of the output classifications and the actual identity, and actual system output. p(i) refers to the identity of a probe image while g(i) refers to the identity of the gallery image with the highest recognition score.

$$P_V = \frac{TP}{TP + FN} \qquad 0 \le P_V \le 1 \tag{14}$$

$$P_F = 1 - \frac{TN}{TN + FP} \qquad 0 \le P_F \le 1 \tag{15}$$

A roc curve is generated by computing P_V and P_F across a range of parameter variables or algorithm variations, each adding a point to the curve. We used two different methods. The first used the ratio of the maximum target layer voltage to the maximum voltage developed during training for that gallery image as our roc curve variable:

$$threshold_{ROC} \propto \frac{V_{\max}^{g(j)} : input = p(k)}{V_{\max}^{g(i)} : input = g(i)} \tag{16}$$

Because of this, the custom target kernels of the gallery set were not normalized as per the method described in Section 3.2.5.

A second curve, ROCIT with DOT, was used by creating a threshold that used information across the entire gallery population. This threshold is a measure of how many standard deviations away a given gallery image's maximum voltage is from the mean of the population:

$$threshold_{ROC\_with\_DOT} \propto \frac{\left[V_{\max}^{g(j)} : input = p(k)\right] - m_{V_{\max}}}{s_{V_{\max}}} \tag{17}$$

where $m_{V_{\max}}$ is the mean of the maximum voltages achieved by all the gallery images for a given probe and $s_{V_{\max}}$ is the standard deviation of the maximum voltages.

If a gallery image's maximum voltage threshold exceeds this value, then the probe is assumed to match the gallery image. Because population statistics are used in the analysis, this method is termed *domain optimized*. In general, this knowledge is not known unless an comparison of a probe against the entire gallery is performed

The parameters used for the algorithm are given in Appendix C and Appendix D, which list the network specification input file and an example custom target kernel file, respectively. Another notable detail was that we scaled down the images to 138 pixels high by 92 pixels wide using bicubic interpolation. This is less than 35% of the original image size. This reduction causes a loss of information, but was necessary to run the benchmark problem in a reasonable amount of time. ROCIT version v1_5 was used for training in the benchmark calculations and SpikeWave version 1.5 was used for recognition..

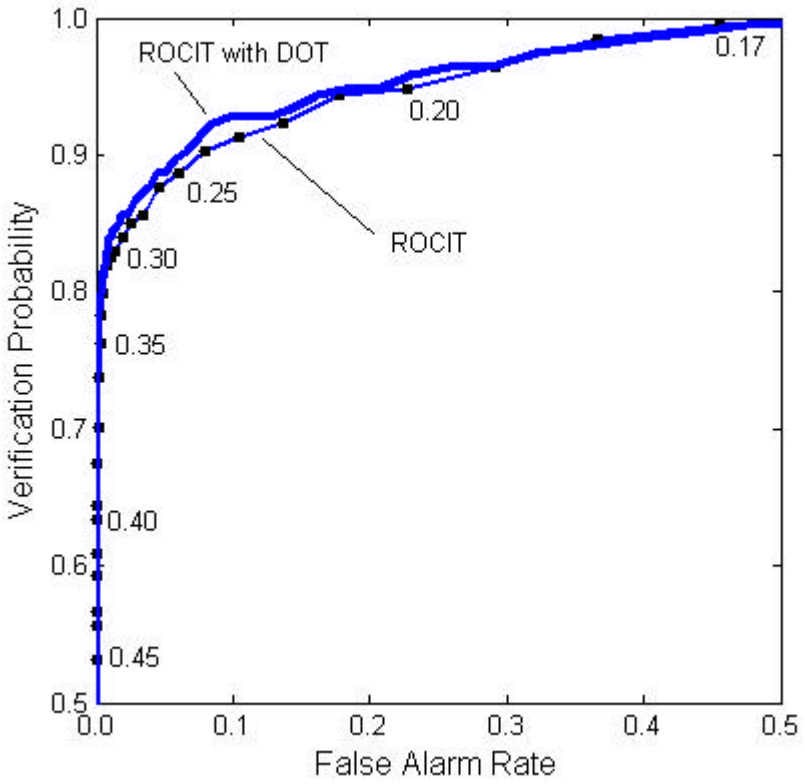


Figure 5-2. Receiver operating characteristic (roc) curve for the baseline ROCIT algorithm. Numbers along the curve represent the voltage ratio thresholds.

5.3 Baseline Algorithm Benchmark Results

Figure 5-2 shows the roc curves for the baseline ROCIT algorithm. Note that the *y*-axis begins at 0.5. Numbers along the lower curve represent voltage ratio thresholds. A 45º roc curve would indicate random chance, and the more area under the curve, the better the algorithm's performance. The baseline ROCIT algorithm achieves approximately 92% true positives and 10% false alarms near a threshold of 0.24. The domain optimized thresholding method (ROCIT with DOT) performs slightly better for false alarm rates below 0.3, but the difference is not dramatic. Figure 5-3 compares ROCIT with results of a number of recognition algorithms from universities and laboratories across the nation[11]. Note that the ARL Correlation curve represents straight-forward cross-correlation, which is outperformed by all other methods for false alarm rates below 0.5.

ROCIT performed quite well against the other algorithms and only the USC algorithm appears to consistently outperform it (except at the smallest false alarm rates). Given that

[11] Performance results for the FERET fafc test have been compiled at: http://www.itl.nist.gov/iad/humanid/feret/perf/score_roc/fafc/, Accessed April, 2004.

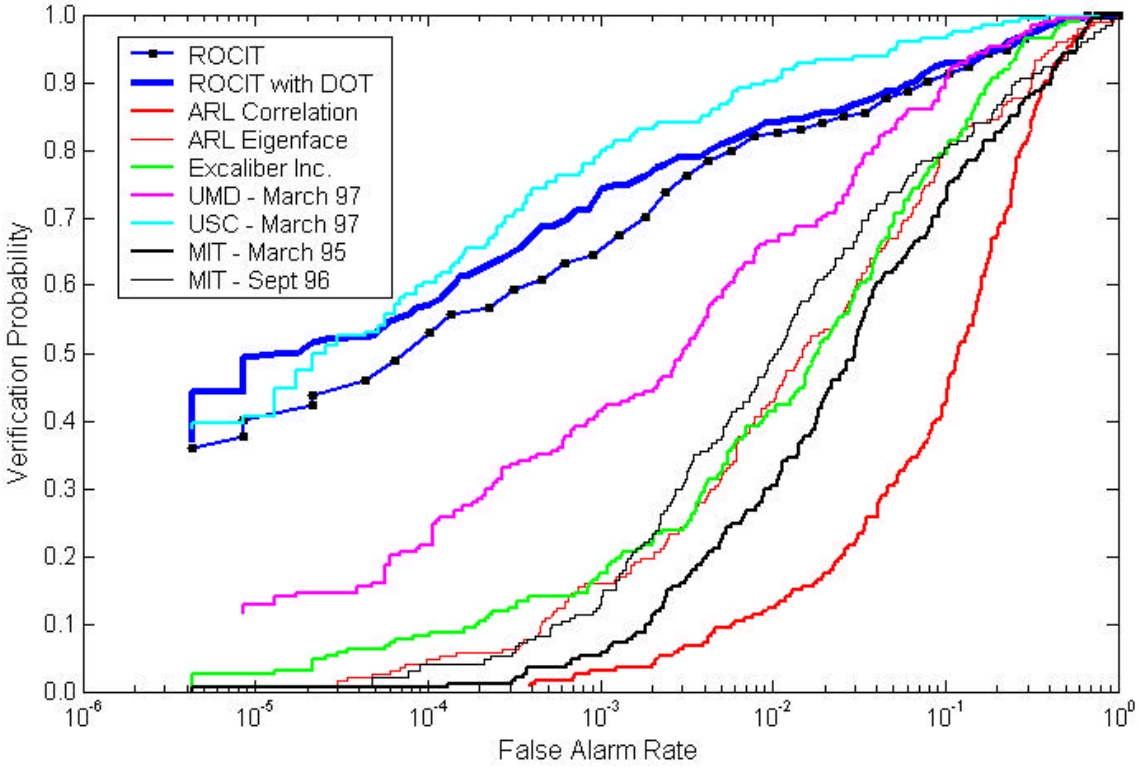


Figure 5-3. Roc curves for various recognition algorithms. ROCIT is the baseline version of the ROCIT algorithm. ROCIT with DOT refers to ROCIT with domain optimized thresholding. Note: The x-axis has been plotted on a log scale.

Recognition Algorithm	Equal Error Rate
USC - March 97	0.051
ROCIT with DOT	***0.082***
ROCIT	***0.093***
UMD - March 97	0.100
Excalibur Corp.	0.145
ARL Eigenface	0.176
MIT - Sept 96	0.180
MIT - March 95	0.193
ARL Correlation	0.248

Table 5-2. Equal error rates of the various algorithms in order from highest to lowest.

Figure 5-4. Examples of mistaken identity from the FERET fafc benchmark using the baseline ROCIT algorithm. The left column is the probe image to be validated. The center column is the training image from the gallery for the same individual as the probe. The right column is the gallery image with the highest score. Note: Numbers indicate scores (ratio of the maximum voltage to the maximum voltage for the training image).

eye coordinates were used by UMD, Excalibur Corp., and both ARL algorithms and that ROCIT used downscaled images, this is encouraging..

It is difficult to rank the algorithms based solely on roc curves because they often overlap at multiple points along the curve. For this reason, roc curves are often compared through a derived quantity known as the equal error rate. The equal error rate is defined as the point at which the false negative rate equals the false positive rate. The lower the equal error rate, the better the algorithm. ROCIT with DOT and ROCIT placed second and third overall with

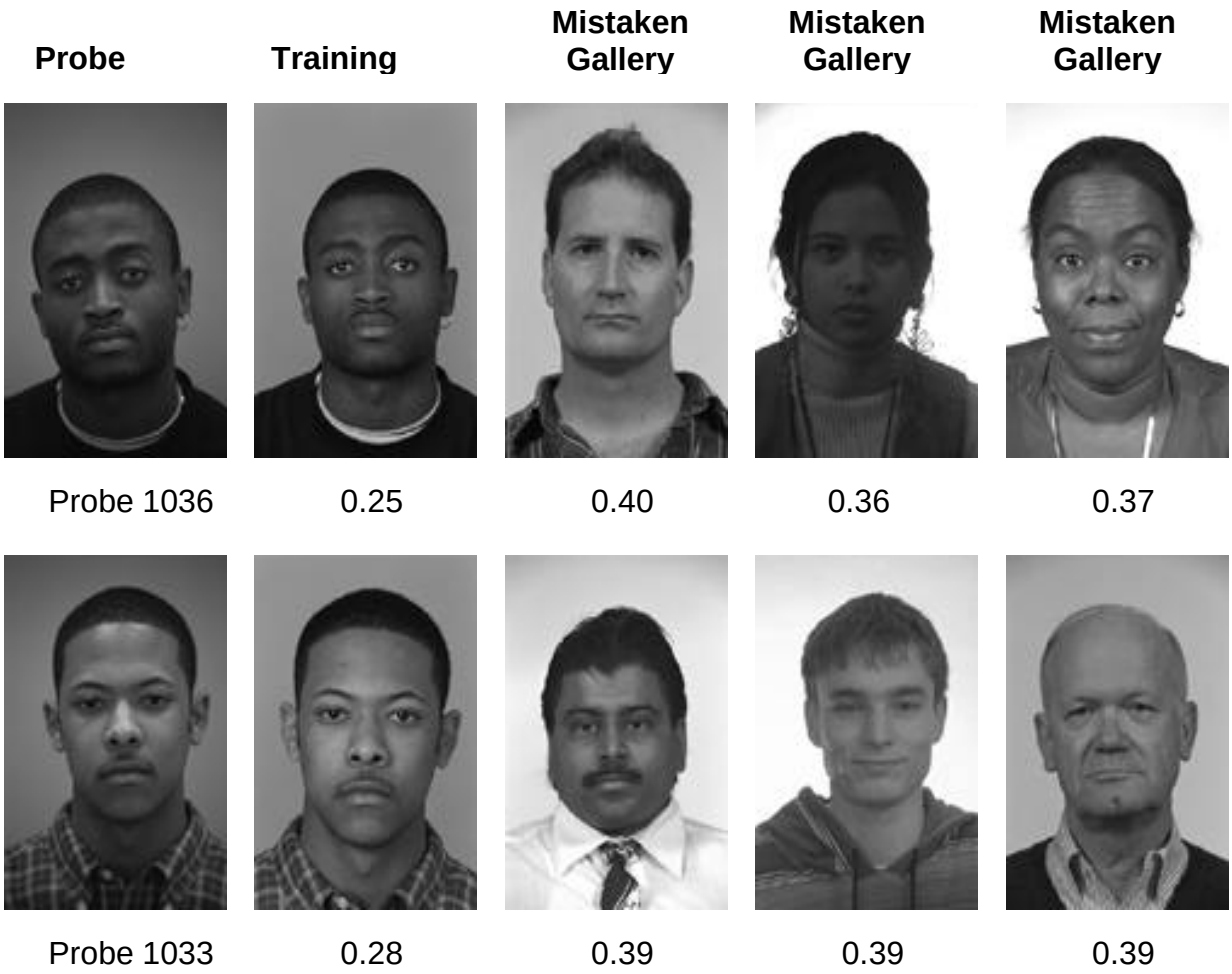


Figure 5-5. A second set of examples of mistaken identity. Both probes are mistaken for several individuals (multiple false positives).

equal error rates of 0.082 and 0.093, respectively. This data is shown in Table 5-2[12]. These results are consistent with the roc curve performance rankings.

It should be emphasized that facial recognition technology has advanced considerably since the FERET benchmarks were developed. However, newer tests no longer provide raw images, just feature vectors, so they are unsuitable for the ROC algorithm approach. Still, the baseline ROCIT algorithm is derived from publications from the late 1990s and so is contemporaneous with the other results. In this light, we can be encouraged with these comparisons and anticipate that modification could continue to improve ROCIT's recognition performance.

[12] Equal error rate results for the FERET fafc test have been compiled at: http://www.itl.nist.gov/iad/humanid/feret/perf/score_roc/score_roc.html, Accessed April, 2004

We have attempted to explore when and why ROCIT failed on some probe images of this benchmark test, but strong conclusions remain elusive. Figure 5-4 shows two examples where ROCIT failed to properly identify a probe image. The top example is puzzling since the training image appears to be very similar in pose, scale, and head orientation. The only differences seem to be lighting and overall contrast. Still, this short haired man was mistakenly identified with a woman who bears almost no resemblance other than her nose. It is easier to explain why the lower example failed to recognize the individual since the training image was at a different scale, but the person who was mistakenly taken to be the probe is very different. Again, this is hard to explain.

Another set of problematic results involving multiple false-positives is shown in Figure 5-5. The upper probe probably fails because of head rotation, and the lower probe is at a different scale and head orientation than the individual's training image. Both probe individuals have distinctive silhouettes as do most of the gallery images that they are mistaken for. These images were run through the ROCIT GUI and the voltage development was followed on a spike by spike basis, and it in no case was it clear exactly which facial or silhouette features of the probe image were aligning with the features of the gallery image target kernels.

An attempt was made to discover fundamental differences between the histogram statistics of probe image and associated false-positive images and those who had been properly identified. These studies were also inconclusive.

6. Potential Algorithm Extensions

The baseline ROCIT algorithm described here has obvious performance weaknesses stemming from its low fidelity representation of the ventral stream. It has only a limited number of neuron types all derived from local brightness contrast. Because of this, it is overly sensitive to brightness contrast in the form of edges or lines. A more detailed model would take advantage of color, gradient, and other textural information in an image.

In line with these issues is the observation that all afferent spike information is "lumped" into a single target kernel. If, during recognition mode, a spike arrives from an afferent position corresponding to a high kernel weight, then a large voltage increase will result, regardless of the selectivity of the afferent layer. A more robust algorithm should result if the target kernel is split into the number of afferent layers. In that way, only spikes arriving with the proper "flavor" will cause a significant voltage change in a target layer. This modification will be essential if color, gradient, or other texture cells are added to the algorithm.

Another limitation of the baseline ROCIT algorithm, as seen in the benchmark results, is its brittleness to changes in in-plane rotation and scale. This probably accounts for a significant portion of the benchmark failures. Scale and rotation invariance could be achieved in one of three ways:

Multiple Target Kernels – Multiple custom target kernels could be developed for the training image at a series of scales and rotations. This method has been implemented in the COTS version of SpikeNET[13]. However, this approach is computationally expensive and would limit a recognition system to a small set of known objects unless significant speed-up could be achieved through a hardware implementation.

Simple Cell / Complex Cell Hierarchy – Several models of object recognition use a hierarchy of "simple" and "complex" cell layers. This approach was pioneered by *Fukishima and Miyake* [1982] in their Neocognitron model that was developed for recognizing handwriting. In this approach, "simple cells" recognize a similar feature, but at with variations (*e.g.*, multiple scales, multiple orientations, *etc.*). "Complex" cells downstream take input from a set of simple cells and produces a single response that is has a degree of invariance to the simple cell variations. How this is accomplished varies between models. *Wallis and Rolls* [1996] emphasize neurologically-inspired competition between neuron groups as do Grossberg and coworkers [*Carpenter et al*., 1989; *Bradski and Grossberg*, 1995]. Most recently, *Reisenhuber and Poggio* [1999] adopted a "max" operator to define response of complex cell layers with moderate success [*Schneider and Riesenhuber*, 2002]. Recent experimental work may bolster this mechanism [*Gawne and Martin*, 2002] (See also *Rousselet et al*. [2003] for a broader discussion these results.). *Reisenhuber and Poggio* [2000] provide a more detailed review of approaches to achieving invariance. Muresan's [*Muresan*, 2002b] hierarchical approach using a rank-order coding framework emulates, to some degree, a max operation. It is essentially a

[13] SNVision, ©2003-2004 SpikeNET Technology, http://www.spikenet-technology.com/, Accessed April, 2004.

race condition between simple cell sets. It is possible that this approach could be extended further.

Foveation With Invariant Transform – A third approach is to sequentially "foveate" on a small regions of an image. At each point of focus, an invariant transformation would be performed upon the image. Options include a log-polar transform as used by *Bradski and Grossberg* [1995] or a combination Fourier-Mellin transform [*Derrode and Ghorbel*, 2001], both of which were successful in recognizing isolated objects against a simple backdrop. *Gonzales* [2003] has confirmed that ROCIT can discriminate faces after they have been log-polar transformed while providing scale and rotational (in-plane) invariance. An example of how log-polar transforms affect images is shown in Figure 6-1. In log-polar space rotation results in translation along the *x*-axis (center and right), while scale variations produce translations along the vertical (not shown). Since ROCIT is already invariant with respect translation, it could recognize scaled and rotated targets anywhere in an image using a log-polar transforms given a center coordinate. This is currently a stumbling block to the approach since it implies that you must know where to look for a target before recognizing it.

The primary challenge to this approach, whether using log-polar transforms or a combination Fourier-Mellin transform is the choice of where to foveate. Each point that is chosen involves a transformation followed by a complete spike propagation computation for the image subset. This is computationally expensive. Random choices could lead to the entire image being sampled. An alternative is to try and use some sort of saliency calculation (*e.g*., *Koch and Ullman* [1985], *Itti* [1998]) at sequentially explore "salient" regions of the image. In general, such calculations are expensive and often require some sort of top-down mechanism.

Li [2002] proposes a saliency map in V1 based on firing rates of feature selective cells. It uses no top-down mechanisms, but appears to be a compelling explanation of fundamental psychoophysics observations including pop-out and segmentation. If a method could be developed to implement this saliency map within a ROC framework, then this could provide a foveation mechanism suitable for ROCIT.

If any of these approaches could be developed and implemented for a rank-order coding framework, then there seems to be a strong chance that ROCIT's recognition performance could improve. Multiple sets of on/off and orientation layers have already been built into the code, so implementing layers having different spatial scales of sensitivity is complete.

Preliminary benchmarking results are encouraging, especially in light of the relative computational simplicity of the method. Future efforts will confront these issues and attempt to explore more complex portions of the problem space of generalized object recognition.

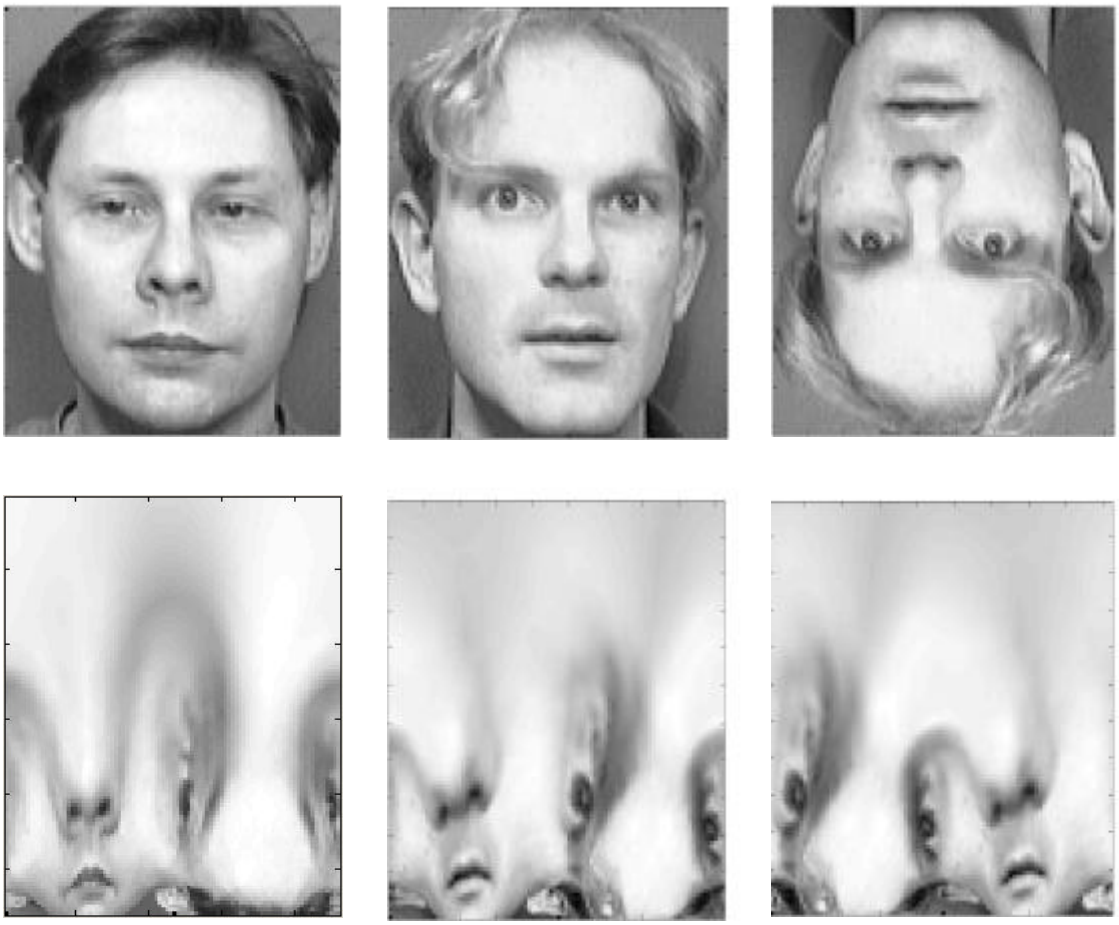


Figure 6-1. Examples of how log-polar transforms affect an image. The top row is the raw image while the bottom is the corresponding transformed image. Rotation results in translation along the x-axis (center and right), while scale variations result in translation along the y-axis (not shown).

This page intentionally left blank.

7. Conclusions

We have implemented a neural-inspired object-recognition algorithm based on a rank-order coding scheme developed by *Gautrais and Thorpe* [1998]. One version, ROCIT was implemented in MATLAB® to enable rapid exploration of algorithm extensions. A second, SpikeWave, has been implemented in C++ for faster execution. Both use interchangeable XML input files and are under strict version control.

Benchmarking against the FERET fafc benchmarking showed better than average performance relative to peers of the same time period (late 1990's). The FERET fafc dataset was developed for biometric discrimination. While ROCIT is intended to serve as a research tool for general object recognition in cluttered, uncontrolled scenes, this benchmark has allowed us to measure how well a rank-order coding scheme handles variable lighting, moderate pose changes, and object morphology plasticity (facial expression changes). These are the first formal benchmark results for such a model.

While these early results are encouraging, the baseline algorithm cannot handle rotation and scale changes beyond a few percent. Scale and rotation are fundamental axes of the problem space of general object recognition, so this issue must be addressed. Several options for modifying the algorithm to accomplish scale and rotation invariance were presented, and others may exist. Given the promising results from the benchmark studies presented here, the baseline algorithm may represent a viable jumping off point for future research into general object recognition in uncontrolled environments.

This page intentionally left blank.

8. References

Biederman, I. (1987). Recognition-by-components: a theory of human image understanding. *Psychological Review*, **94**, 115-147.

Bradski, G., and S. Grossberg (1995). Fast-learning VIEWNET architectures for recognizing three-dimensional objects from multiple two-dimensional views. *Neural Networks*, **8**(7/8), 1053-1080.

Carpenter, G.A., and S. Grossberg (1987a). Invariant pattern recognition and recall by an attentive self-organizing ART architecture in a nonstationary world. In M. Caudill and C. Butler (Eds.), Proceedings of the IEEE International Conference on Neural Networks, 11 (pp. 737-745), New York: IEEE.

Carpenter, G.A., and S. Grossberg (1987b). ART 2: Self-organization of stable category recognition codes for analog input patterns. *Applied Optics*, **26**, 4919-4930.

Carpenter, G.A., S. Grossberg, and C. Mehanian (1989). Invariant recognition of cluttered scenes by a self-organizing ART architecture: CORT-X boundary segmentation, *Neural Networks*, **2**, 169-181.

Delorme, A., J. Gautrais, R. Van Rullen, and S. Thorpe (1999). SpikeNET: A simulator for modeling large networks of integrate and fire neurons. *Neurocomputing*, **26-27**, 989-996.

Delorme, A., Thorpe, S. (2001a). Face processing using one spike per neuron: resistance to image degradation. *Neural Networks*, **14**(6-7), 795-804.

Delorme, A., and S. Thorpe (2001b). Event-driven simulation of large networks of spiking neurons. Unpublished Manuscript, September 15, 2001, 20 pp..

Derrode, S.,a nd F. Ghorbel (2001). Robust and efficient Fourier-Mellin transform approximations for gray-level image reconstruction and complete invariant description. *Computer Vision and Image Understanding*, 83, 57-78.

Edelman, S. (1999) Representation and Recognition in Vision. The MIT Press: Cambridge, Massachusetts, 335 pp..

Felleman, D.J., and D.C. Van Essen (1991). Distributed hierarchical processing in the primate cerebral cortex. *Cerebral Cortex*, **1**(1), 1-47.

Fukishima, K. (2003). Neocognitron for handwritten digit recognition. *Neurocomputing*, **51**, 161-180.

Fukishima, K. (1980). Neocognitron: a self-organizing neural network for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, **36**, 193-202.

Fukushima, K., and S. Miyake (1982). Neocognitron: a new algorithm for pattern recognition tolerant of deformations and shifts in position. *Pattern Recognition*, **15**(6), 455-469.

Gautrais, J., and S. Thorpe (1998). Rate coding versus temporal order coding: a theoretical approach. *Biosystems*, **48**, 57-65.

Gawne, T.J., and J.M. Martin (2002). Responses of primate visual cortical V4 neurons to simultaneously presented stimuli. *Journal of Neurophysiology*, **88**, 1128-1135.

Gonzales, A.I. (2003) Log-Polar Transforms: Rotation and Scale Invariance for ROCIT? Presentation to Sandia National Laboratories, October 16, 2003.

Grill-Spector, K. (2003). The neural basis of object perception. *Current Opinion in Neurobiology*, **13**, 1-8.

Grill-Spector, K., T. Kushnir, S. Edelman, G. Avidan, Y. Itzchak, and R. Malach (1999). Differential processing of objects under various viewing conditions in the human lateral occipital complex. *Neuron*, **24**, 187-203.

Grill-Spector, K., Z. Kourtzi, and N. Kanwisher (2001). The lateral occipital complex and its role in object recognition. *Vision Research*, **41**, 1409-1422.

Grossberg, S. (2003). Laminar cortical dynamics of visual form perception. *Neural Networks*, **16**, 925-931.

Grossberg, S., and G. Bradski (1995) VIEWNET architectures for invariant 3-D object learning and recognition from multiple 2-D views. In B. Bouchon-Meunier, R. Yager and L. Azdeh (eds.), Fuzzy Logic and Soft Computing. Singapore: World Scientific Publishing.

Hubel, D.H., and T.N. Wiesel (1977). Functional architecture of macaque monkey visual cortex. *Proceedings of the Royal Society of London, Series B*, **198**, 1-59.

Itti, L., C. Koch, and E. Niebur, (1998). A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions in Pattern Analysis and Machine Intelligence*, **20**(11), 1254-1259.

Koch, C., and S. Ullman (1985). Shifts n selective visual attention: towards the underlying neural circuitry. *Human Neurobiology*, **4**, 219-227.

Lamme, VAF, and P.R. Roelfsema (2000). The distinct modes of vision offered by feedforward and recurrent processing. *Trends in Neurosciences*, **23**(11), 571-579.

Li, F.F., R. Van Rullen, C. Koch, and P. Perona (2002). Rapid natural scene categorization in the near absence of attention. *Proc. Natl. Acad. Sci*., **99**, 9596-9601.

Li, Z. (2002). A saliency map in primary visual cortex. *Trends in Cognitive Science*, **6**(1), 9-16.

Logothetis, N.K., J. Pauls and T. Poggio (1995). Shape representation in the inferior temporal cortex of monkeys. *Current Biology*, **5**, 552-563.

Logothetis, N.K., J. Pauls, H.H. Bülthoff, and T. Poggio (1995). View-dependent object recognition by monkeys. *Current Biology*, **4**(**5)**, 401-414.

Logothetis, N.K. and J. Pauls (1995). Psychophysical and physiological evidence for viewer-centered object representations in the primate. *Cerebral Cortex*, **3**, 270-288.

Logothetis, N., and D. Sheinberg (1996). Visual object recognition, *Annual Review of Neuroscience*, **19**, 577-621.

Martin, K.A.C. (2002). Microcircuits in visual cortex, *Current Opinion in Neurobiology*, **12**(4), 418-425.

Muresan, R.C. (2003). RetinotopicNET: An Efficient Simulator for Retinotopic Visual Architectures. Proceedings of the European Symposium on Artificial Neural Networks, Bruges, Belgium, pp 247-254.

Muresan, R.C. (2002a). Complex Object Recognition Using a Biologically Plausible Neural Model. In: Advances in Simulation, Systems Theory and Systems Engineering, WSEAS Press: Athens, pp. 163-168.

Muresan, R.C. (2002b). Visual Scale Independence in a Network of Spiking Neurons. In: Proceedings of the 9th International Conference on Neural Information Processing, Singapore 18-22 Nov. 2002, IEEE, Vol. 4, pp 1739-1743.

Oram., M.W., and D.I. Perret (1992). Time course of neural responses discriminating different views of the face and head. *Journal of Neurophysiology*, **68**(1), 70-84.

Phillips, P.J., P.J. Rauss, and S.Z Der (1996) FERET (Face Recognition Technology) Recognition Algorithm Development and Test Results, Army Research Laboratory Technical Report, ARL-TR-995, 73 pp.

Poggio, T., and S. Edelman (1990). A network that learns to recognize three-dimensional objects. *Nature*, **343**, 263-266.

Raizada, R.D.S., and S. Grossberg (2003). Towards a theory of the laminar architecture of cerebral cortex: computational clues from the visual system. *Cerebral Cortex*, **13**, 100-113.

Regan, D. (2000). Human Perception of Objects: Early Visual Processing of Spatial Form Defined by Luminance, Color Texture, Motion, and Binocular Disparity. Sinauer Associates, Inc., Publishers: Sunderland, Massachusetts, 577 pp.

Riesenhuber, M., and T. Poggio (2000). Computational Models of Object Recognition in Cortex: A Review. Massachusetts Institute of Technology, AI Memo 1695, CBCL Paper No. 190, 10 pp..

Riesenhuber, M., and T. Poggio (1999). Hierarchical models of object recognition in cortex. *Nature Neuroscience*, **2**(11), 1019-1025.

Rizvi, D.S., P. J. Phillips and H. Moon. (1998), The FERET Verification Testing Protocol for Face Recognition Algorithms. National Institute of Standards and Technology Internal Report, NISTIR 6281, 16 pp.

Rolls, E.T., and G. Deco (2002). Computational Neuroscience of Vision. Oxford University Press, 569 pp.

Rolls, E.T., and T. Milward (2000). A model of invariant object recognition in the visual system: learning rules, activation functions, lateral inhibition, and information-based performance measures. *Neural Computation*, **12**, 2547-2572.

Rousselet, G.A., S.J. Thorpe, and M. Fabre-Thorpe (2003). Taking the MAX from neuronal responses. *Trends in Cognitive Sciences*, **7**(3), 99-102.

Schneider, R., and M. Reisnenhuber (2002). A Detailed Look at Scale and Translation Invariance in a Hierarchical Neural Model of Visual Object Recognition. Massachusetts Institute of Technology, AI Memo 2002-011, CBCL Memo 218, 13 pp..

Tanaka, K. (1996). Inferotemporal cortex and object vision. *Annual Reviews of Neuroscience*, **19**, 109-139.

Tanaka, Y., H. Saito, Y. Fukada, and M. Moriya (1991). Coding visual images of objects in the inferotemporal cortex of the macaque monkey. *Journal of Neurophysiology*, 66, 170-189.

Thorpe, S., A. Delorme, and R. Van Rullen (2001). Spike-based strategies for rapid processing. *Neural Netw*orks (**14**), 715-725.

Thorpe, S., and J. Gautrais (1998). Rank Order Coding. In Computational Neuroscience: Trends in Research 1998, J. Bower Editor, Plenum Press: New York, pp. 113-118.

Thorpe, S., A. Delorme, and R. Van Rullen (2001). Spike-based strategies for rapid processing. *Neural Networks*, **14**, 715-725.

Thorpe, S., D. Fize, and C. Marlot (1996). Speed of processing in the human visual system. *Nature*, **381**, 520-522.

Ungerleider, L.G., and J.V. Haxby (1994). 'What' and 'where' in the human brain. *Current Opinion in Neurobiology*, **4**, 157-165.

Van Rullen, R., (2003). Visual saliency and spike timing in the ventral visual pathway. Journal of Physiology - Paris, 97(2-3), 365-377.

Van Rullen, R., J. Gautrais, A. Delorme, and S. Thorpe (1998). Face processing using one spike per neurone, *Biosystems*, **48**, 229-239.

Wallis, G., and E.T. Rolls (1997). A model of invariant object recognition in the visual system. *Progress in Neurobiology*, **51**, 167-194.

Wyeth, B. (1999). Spike timing in the mammalian visual system. *Current Opinion in Neurobiology*, **9**, 447-453.

This page intentionally left blank.

Appendix A: Algorithmic Complexity Analysis

In this derivation the following assumptions are made:

- The times required to perform any individual operation are assumed to be the same (addition, subtraction, multiplication, division, comparison, assignment, etc.).
- Pairs of kernels are stored for each layer to eliminate the need to multiply by a bias value during voltage, sensitivity, and target kernel updates.

Notation is as follows:

N	Number of pixels in an input image
G	Number of groups of input and orientation layers (recall that each set of on/off layers is paired with a group of eight orientation layers)
T	Number of target layers (*i.e.*, number of targets sought)
W_i^{in}	Number of weights in the kernel of the *i*th group of input (*in*) layers
W_i^{or}	Number of weights in the kernel of the *i*th group of orientation (*or*) layers
W_j^t	Number of weights in the kernel of target layer *j*
f_i^{in}	Fraction of spikes propagated from *i*th input layer group, $0 < f_i^{in} \leq 1$
f_i^{or}	Fraction of spikes propagated from *i*th orientation layer group, $0 < f_i^{or} \leq 1$
A	Time required to perform a single mathematical operation (*i.e.*, addition, subtraction, multiplication, division, comparison, *etc*.)

The algorithmic complexity of the ROCIT baseline algorithm can be computed by examining the computational work associated with each layer and by considering how the algorithm changes between training and recognition modes.

Algorithmic Complexity of the On/Off Layer Preprocess

Each on/off layer undergoes a convolution followed by sorting as a way of ranking spikes (See Section 3.2.1). The convolution consists of moving the kernel across the image and performing a point by point multiplication, summing the values, and assigning them to a new, intermediate matrix. This is a conservative estimate since, for symmetric convolution kernels, this can be reduced. This resultant matrix is then sorted, which can be performed

$O(N \log N)$ time (*e.g.*, merge sort or heap sort). Therefore, work required to process G pairs of on/off layers is

$$C^{in} = \sum_{i=1}^{G} 2\left[2AW_i^{in} N + BN \log N\right] \quad \text{(A1)}$$

Here, the first 2 accounts for the on/off pairs and the second 2 accounts for the multiplications and additions associated with the convolution. The second term in brackets is the time complexity of the sorting algorithm ($N \log N$) and B is its constant.

Algorithmic Complexity of the Orientation Layer Updates

For each spike that fires in an on/off layer group, a local update is performed in all eight layers of its associated orientation-layer group. First the voltage is updated, then the sensitivity is updated, and finally, the voltage threshold is checked for new spikes (See Equation (6), (12). The updates are all local and involve an area the size of the orientation layer kernel centered around the afferent spike location. Voltage and sensitivity updates require a multiply and addition per kernel weight, and thresholding requires one comparison per kernel weight and an assignment for every threshold excursion. A conservative assumption is that every spike causes every voltage in the affected area to exceed the threshold, then the work required to update the orientation layers is thus:

$$C^{or} = \sum_{i=1}^{G} f_i^{in} N 8\left[\underbrace{2AW_i^{or}}_{Voltage} + \underbrace{2AW_i^{or}}_{Sensivity} + \underbrace{2AW_i^{or}}_{Threshold}\right] \quad \text{(A2)}$$

where $f_i^{in} N$ is the total number of neurons firing from the *i*th input-layer group.

Algorithmic Complexity of the Custom Target Kernel Updates

Target kernel updates are based on Equation (8). They involve multiplications and additions over an area the size of the *afferent* layer kernel. Global desensitization is used during training and no voltage threshold checking takes place, so the target kernel updates represent the vast majority of work for the target layers during training. This gives a time complexity of:

$$C^{k} = \sum_{i=1}^{G} f_i^{or} N \underbrace{2AW_i^{or}}_{Kernel} \quad \text{(A3)}$$

where $f_i^{or} N$ is the total number of neurons firing from the *i*th orientation-layer group.

Algorithmic Complexity of the Target Layer Updates

Target layer updates involve the same operations as orientation layers, but with afferent spikes coming from orientation layers and the custom target kernel of the respective target layer used for updates (See Equation). Therefore, the work associated with target layer updates is:

$$C^t = \sum_{j=1}^{T}\sum_{i=1}^{G} f_i^{or} N8\left[\underbrace{2AW_j^t}_{Voltage} + \underbrace{2AW_j^t}_{Sensivity} + \underbrace{2AW_j^t}_{Threshold}\right] \quad \text{(A4)}$$

where f_i^{or} is the total number of neurons firing from the *i*th orientation-layer group.

Algorithmic Complexity of Training

The algorithmic complexity of training amounts to:

$$C^{tr} = C^{in} + C^{or} + C^k \quad \text{(A5)}$$

or,

$$C^{tr} = \sum_{i=1}^{G} 2\left[2AW_i^{in} N + BN\log N\right] + \sum_{i=1}^{G} f_i^{in} 48AW_i^{or} N + \sum_{i=1}^{G} f_i^{or} 2AW_i^{or} N \quad \text{(A6)}$$

after substitution of Equations (A1), (A2), and (A3) and rearranging. If we assume that the size of the input layer and orientation layer kernels are small, relative to the size of the target image, then we can ignore the differences in their individual sizes such that:

$$W_i^{in} \approx W_{i+1}^{in} = W^{in} << N \quad \text{(A7)}$$

$$W_i^{or} \approx W_{i+1}^{or} = W^{or} << N\,. \quad \text{(A8)}$$

If we further ignore the differences in sizes between the input layer kernels and the orientation layer kernels, then we can approximate the number of weights as

$$W^{in} \approx W^{or} = \tilde{W} \quad \text{(A9)}$$

If we further assume that each group of layers of a given type (input or orientation) propagates a similar fraction of spikes, then we can write

$$f_i^{in} \approx f_{i+1}^{in} = \tilde{f}^{in} \quad \text{(A10)}$$

$$f_i^{or} \approx f_{i+1}^{or} = \tilde{f}^{or}\,. \quad \text{(A11)}$$

Given these assumptions, Equation (A6) reduces to

$$C^{tr} \approx 2GA\tilde{W}\left[\left(24\tilde{f}^{in} + \tilde{f}^{or} + 2\right)N + CN\log N\right] \quad \text{(A12)}$$

where $C = B/A\tilde{W}$.

This expression indicates that training is $O(N\log N)$ as long as

$$2GA\tilde{W}\left(24\tilde{f}^{in}+\tilde{f}^{or}+2\right)<<N. \tag{A13}$$

Therefore, if the number of groups, *G*, is large, the size of the kernels is large, or if a high fraction of spikes is required to propagate for the development of a useful target kernel, then the algorithm can approach $O\left(N^2\right)$.

Algorithmic Complexity of Recognition

The algorithm complexity of recognition is given by:

$$C^{rec}=C^{in}+C^{or}+C^{t} \tag{A14}$$

or,

$$C^{rec}=\sum_{i=1}^{G}2\left[2AW_i^{in}N+BN\log N\right]+\sum_{i=1}^{G}f_i^{in}48AW_i^{or}N+\sum_{j=1}^{T}\sum_{i=1}^{G}f_i^{or}N48AW_j^{t} \tag{A15}$$

Making the same assumptions as before regarding the relative sizes of the input and orientation layer kernels and fractions of spikes propagating from input and orientation layers (Equations A7-A11), and further assuming that the targets are similar in size, *i.e.*,

$$W_j^t\approx W_{j+1}^t=\tilde{W}^t \tag{A16}$$

then Equation (A15) may be rewritten as

$$C^{rec}\approx 2GA\tilde{W}\left[\left(24\tilde{f}^{in}+\tilde{f}^{or}+24\tilde{f}^{or}T\frac{\tilde{W}^t}{\tilde{W}}\right)N+CN\log N\right]. \tag{A17}$$

Thus, recognition is also of $O\left(N\log N\right)$ as long as

$$2GA\tilde{W}\left(24\tilde{f}^{in}+\tilde{f}^{or}+24\tilde{f}^{or}T\frac{\tilde{W}^t}{\tilde{W}}\right)<<N. \tag{A18}$$

This constraint may be violated as the number of groups increases, the size of the input and orientation layers increase, or the size of the target kernels approach the size of the image being processed, *i.e.*, whenever $\tilde{W}^t\approx N$. This will be the case when, for example, "mug shots" are being processed in which case all images may be approximately the same number of pixels. In this case, the algorithm becomes $O\left(N^2\right)$.

If the targets are hidden in a much larger image, then $\tilde{W}\approx\tilde{W}^t<<N$. Likewise, the fraction of spikes propagating from the on/off layers may approach one since the target may be low in

contrast relative to other features in the image. If we assume that the fraction of neurons that fire in orientation layers is 0.5[14], then the work required for recognition becomes

$$C^{rec} \approx GA\tilde{W}\left[(49+24T)N + 2CN\log N\right], \tag{A19}$$

which is still $O(N\log N)$ while

$$GA\tilde{W}(49+24T) << N\ . \tag{A20}$$

Consider a single target that measures 100x100 pixels, hidden in a five megapixel image and a network that consists of a single group of input and orientation layers. In this case, Equation (A19) becomes

$$C^{rec} \approx (A\cdot 0.0146\,N)N + (B\cdot 0.00000268\ N)N \qquad (N = 5{,}000{,}000) \tag{A21}$$

whose first term is marginally less than $O(N^2)$. It is clear to see, that as the network grows in complexity, or if the target images are any larger with respect to the input image, then Equation (A19) quickly approaches $O(N^2)$ despite the fact that the input and orientation layer calculations are amortized across a set of target layers.

[14] This is a conservative assumption. Experience has indicated that the ratio of input layer spikes to orientation layer spikes is typically less than one fifth.

This page intentionally left blank.

Appendix B: XML Input File Element Tags

ROOT	Root XML Level

Network Architecture Parameters

NUMLAYERS	Number of Layers
CONNECTIONS	Layer Connection Matrix

Initial Parameter Settings

TH_INIT	Initial Voltage Thresholds of Layers
V_INIT	Initial Voltages of Layers
S_INIT	Initial Sensitivities of Layers
IMAGE_SCALE	Relative Layer Scale (1 = Original Scale of Image)
BIAS	Layer Bias Values
GROUP	Layer Group Affiliations
GROUP_NAME	Group Names
TH_INIT_TARGET	Initial Target Layer Voltage Threshold - Recognition Mode
V_INIT_TARGET	Initial Target Layer Voltage - Recognition Mode
S_INIT_TARGET	Initial Target Layer Sensitivity - Recognition Mode
TH_INIT_TARGET_LEARN	Initial Target Layer Voltage Threshold – Training Mode
V_INIT_TARGET_LEARN	Initial Target Layer Voltage – Training Mode
S_INIT_TARGET_LEARN	Initial Target Layer Sensitivity – Training Mode
SENSITIVITY_FLAG_LEARN	Flag: 1 = Global Sensitivity Update - Training Mode 2 = Local Sensitivity Update - Training Mode
SENSITIVITY_FLAG_RECOG	Flag: 1 = Global Sensitivity Update - Recognition Mode 2 = Local Sensitivity Update - Recognition Mode
IMAGE_SCALE_TARGET	*Currently Unused*
BIAS_TARGET	*Currently Unused*
NORMALIZE	Flag: Kernel Normalization 1 = Normalize Kernels to a Fixed Amplitude 2 = Normalized Kernels to Achieve Max Voltage = 1.
READ_KERNELS	Flag: (LoG and Gabor Kernels Only) 1 = Read From File 0 = Compute From Parameter List
MAXTIME	Maximum Time Step

Kernel Specifications

KERNEL	Kernel Parameter Specifications
FILENAME	Name of File if READ_KERNELS=1
LAYER	Layer Assignment
KWIDTH	Kernel Width
KHEIGHT	Kernel Height
NAME	Kernel Name
TYPE	Kernel Type (Gabor or LoG)
FREQUENCY	Spatial Frequency (Gabor Only)
ALIGNMENT	Kernel Orientation (Gabor Only)
SIGMA	Gaussian Smoothing Parameter
PHASE	Phase of Planar Wave (Gabor Only)
MEAN	Mean Kernel Value
AMPLITUDE	Maximum Kernel Amplitude for Normalization

Custom Target Specifications

TARGET	Custom Target Kernel Tag
MAX_VOLTAGE	Maximum Voltage for Achieved During Training
TARGET_PATTERN_W	Voltage Pattern Width
TARGET_PATTERN_H	Voltage Pattern Height
PATTERN_CONV_MAX	Maximum Cross-Correlation Value for Voltage Pattern
TARGET_PATTERN	Voltage Pattern Matrix
NAME	Target Name
WIDTH	Kernel Matrix Width
HEIGHT	Kernel Matrix Height
CURRENT_TIME	Training Steps Used
CUSTOM	Target Kernel Matrix
IMAGE	Raw Image of Target

Appendix C: Example Network Specifications File

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<ROOT>
<NUMLAYERS>10</NUMLAYERS>
<NAME>Default Network</NAME>
<CONNECTIONS>
0 0 1 1 1 1 1 1 1 1
0 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
</CONNECTIONS>

<TH_INIT>
0.15 0.15 2.500000e+000 2.500000e+000 2.500000e+000 2.500000e+000 2.500000e+000
   2.500000e+000 2.500000e+000 2.500000e+000
</TH_INIT>

<V_INIT>
0 0 0 0 0 0 0 0 0 0
</V_INIT>

<S_INIT>
1 1 1 1 1 1 1 1 1 1
</S_INIT>

<IMAGE_SCALE>
1 1 1 1 1 1 1 1 1 1
</IMAGE_SCALE>

<BIAS>
1 -1 1 1 1 1 -1 -1 -1 -1 1
</BIAS>

<GROUP>
1 1 2 2 2 2 2 2 2 2 3
</GROUP>

<GROUP_NAME>
On/Off
Edges
Target
</GROUP_NAME>

<TH_INIT_TARGET> 50000000 </TH_INIT_TARGET>

<V_INIT_TARGET> 0  </V_INIT_TARGET>

<S_INIT_TARGET> 0.9999 </S_INIT_TARGET>

<TH_INIT_TARGET_LEARN> 50000000 </TH_INIT_TARGET_LEARN>

<V_INIT_TARGET_LEARN> 0 </V_INIT_TARGET_LEARN>

<S_INIT_TARGET_LEARN> 0.9999 </S_INIT_TARGET_LEARN>

<IMAGE_SCALE_TARGET> 1 </IMAGE_SCALE_TARGET>
```

```
<CONNECTION_TO_TARGET>
0 0 1 1 1 1 1 1 1 1
</CONNECTION_TO_TARGET>

<CONNECTION_FROM_IMAGE>
1 1 0 0 0 0 0 0 0 0
</CONNECTION_FROM_IMAGE>

<SENSITIVITY_FLAG_LEARN> 1 </SENSITIVITY_FLAG_LEARN>
<SENSITIVITY_FLAG_RECOG> 1 </SENSITIVITY_FLAG_RECOG>

<BIAS_TARGET> 1 </BIAS_TARGET>

<NORMALIZE> 2 </NORMALIZE>

<READ_KERNELS> 1 </READ_KERNELS>

<MAXTIME> 500 </MAXTIME>


<KERNEL>
  <FILENAME> \\Soma\1\Dev\SpikeWave\bdfarka_030504\spikewave\resources\
   on_kernel_5_x_5.txt </FILENAME>
  <LAYER>1</LAYER>
  <KWIDTH>5</KWIDTH>
  <KHEIGHT>5</KHEIGHT>
  <NAME>On</NAME>
  <TYPE>log</TYPE>
  <PHASE>-1</PHASE>
  <SIGMA>5.000000e-001</SIGMA>
  <DIM>5</DIM>
</KERNEL>

<KERNEL>
  <FILENAME> \\Soma\1\Dev\SpikeWave\bdfarka_030504\spikewave\resources\
  off_kernel_5_x_5.txt </FILENAME>
  <LAYER> 2 </LAYER>
  <KWIDTH> 5 </KWIDTH>
  <KHEIGHT> 5 </KHEIGHT>
  <PHASE> 1 </PHASE>
  <NAME> Off </NAME>
  <TYPE> log </TYPE>
  <SIGMA> 5.000000e-001 </SIGMA>
  <DIM> 5 </DIM>
</KERNEL>

<KERNEL>
  <FILENAME> \\Soma\1\Dev\SpikeWave\bdfarka_030504\spikewave\resources\
  orient_000_15_x_15.txt</FILENAME>
  <LAYER> 3 </LAYER>
  <KWIDTH> 15 </KWIDTH>
  <KHEIGHT> 15 </KHEIGHT>
  <NAME> 0 Degrees </NAME>
  <TYPE> gabor </TYPE>
  <FREQUENCY> 15 </FREQUENCY>
  <ALIGNMENT> 0 </ALIGNMENT>
  <SIGMA> 2.500000e+000 </SIGMA>
  <PHASE> 90 </PHASE>
  <MEAN> 0 </MEAN>
  <AMPLITUDE>1 </AMPLITUDE>
</KERNEL>
```

```
<KERNEL>
  <FILENAME> \\Soma\1\Dev\SpikeWave\bdfarka_030504\spikewave\resources\
  orient_045_15_x_15.txt </FILENAME>
  <LAYER> 4 </LAYER>
  <KWIDTH> 15 </KWIDTH>
  <KHEIGHT> 15 </KHEIGHT>
  <NAME> 45 Degrees </NAME>
  <TYPE> gabor </TYPE>
  <FREQUENCY> 15 </FREQUENCY>
  <ALIGNMENT> 45 </ALIGNMENT>
  <SIGMA> 2.500000e+000 </SIGMA>
  <PHASE> 90 </PHASE>
  <MEAN> 0 </MEAN>
  <AMPLITUDE>1</AMPLITUDE>
</KERNEL>

<KERNEL>
  <FILENAME> \\Soma\1\Dev\SpikeWave\bdfarka_030504\spikewave\resources\
  orient_090_15_x _15.txt </FILENAME>
  <LAYER> 5 </LAYER>
  <KWIDTH> 15 </KWIDTH>
  <KHEIGHT> 15 </KHEIGHT>
  <NAME> 90 Degrees </NAME>
  <TYPE> gabor </TYPE>
  <FREQUENCY> 15 </FREQUENCY>
  <ALIGNMENT> 90 </ALIGNMENT>
  <SIGMA> 2.500000e+000 </SIGMA>
  <PHASE> 90 </PHASE>
  <MEAN> 0 </MEAN>
  <AMPLITUDE> 1 </AMPLITUDE>
</KERNEL>

<KERNEL>
  <FILENAME> \\Soma\1\Dev\SpikeWave\bdfarka_030504\spikewave\resources\
  orient_135_15_x _15.txt </FILENAME>
  <LAYER> 6 </LAYER>
  <KWIDTH> 15 </KWIDTH>
  <KHEIGHT> 15 </KHEIGHT>
  <NAME> 135 Degrees </NAME>
  <TYPE> gabor </TYPE>
  <FREQUENCY> 15 </FREQUENCY>
  <ALIGNMENT> 135 </ALIGNMENT>
  <SIGMA> 2.500000e+000 </SIGMA>
  <PHASE> 90 </PHASE>
  <MEAN> 0 </MEAN>
  <AMPLITUDE>1 </AMPLITUDE>
</KERNEL>

<KERNEL>
  <FILENAME> \\Soma\1\Dev\SpikeWave\bdfarka_030504\spikewave\resources\
  orient_180_15_x_15.txt </FILENAME>
  <LAYER> 7 </LAYER>
  <KWIDTH> 15 </KWIDTH>
  <KHEIGHT> 15 </KHEIGHT>
  <NAME> 180 Degrees </NAME>
  <TYPE> gabor </TYPE>
  <FREQUENCY> 15 </FREQUENCY>
  <ALIGNMENT> 180 </ALIGNMENT>
  <SIGMA> 2.500000e+000 </SIGMA>
  <PHASE> 90 </PHASE>
  <MEAN> 0 </MEAN>
  <AMPLITUDE> 1 </AMPLITUDE>
</KERNEL>
```

```
<KERNEL>
  <FILENAME> \\Soma\1\Dev\SpikeWave\bdfarka_030504\spikewave\resources\
  orient_225_15_x _15.txt </FILENAME>
  <LAYER> 8 </LAYER>
  <KWIDTH> 15 </KWIDTH>
  <KHEIGHT> 15 </KHEIGHT>
  <NAME> 225 Degrees </NAME>
  <TYPE> gabor </TYPE>
  <FREQUENCY> 15 </FREQUENCY>
  <ALIGNMENT> 225 </ALIGNMENT>
  <SIGMA> 2.500000e+000 </SIGMA>
  <PHASE> 90 </PHASE>
  <MEAN> 0 </MEAN>
  <AMPLITUDE> 1 </AMPLITUDE>
</KERNEL>

<KERNEL>
  <FILENAME> \\Soma\1\Dev\SpikeWave\bdfarka_030504\spikewave\resources\
  orient_270_15_x _15.txt </FILENAME>
  <LAYER> 9 </LAYER>
  <KWIDTH> 15 </KWIDTH>
  <KHEIGHT> 15 </KHEIGHT>
  <NAME> 270 Degrees </NAME>
  <TYPE> gabor </TYPE>
  <FREQUENCY> 15 </FREQUENCY>
  <ALIGNMENT> 270 </ALIGNMENT>
  <SIGMA> 2.500000e+000 </SIGMA>
  <PHASE> 90 </PHASE>
  <MEAN> 0 </MEAN>
  <AMPLITUDE> 1 </AMPLITUDE>
</KERNEL>

<KERNEL>
  <FILENAME> \\Soma\1\Dev\SpikeWave\bdfarka_030504\spikewave\resources\
  orient_315_15_x _15.txt </FILENAME>
  <LAYER> 10 </LAYER>
  <KWIDTH> 15 </KWIDTH>
  <KHEIGHT> 15 </KHEIGHT>
  <NAME> 315 Degrees </NAME>
  <TYPE> gabor </TYPE>
  <FREQUENCY> 15 </FREQUENCY>
  <ALIGNMENT> 315 </ALIGNMENT>
  <SIGMA> 2.500000e+000 </SIGMA>
  <PHASE> 90 </PHASE>
  <MEAN> 0 </MEAN>
  <AMPLITUDE> 1 </AMPLITUDE>
</KERNEL>

</ROOT>
```

Appendix D: Example Custom Target File

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!DOCTYPE note SYSTEM "target.dtd">

<ROOT>

<TARGET>

<MAX_VOLTAGE> 2.6832 </MAX_VOLTAGE>

<TARGET_PATTERN_W> 17 </TARGET_PATTERN_W>

<TARGET_PATTERN_H> 25 </TARGET_PATTERN_H>

<PATTERN_CONV_MAX> -1.439488e+001 </PATTERN_CONV_MAX>

<TARGET_PATTERN>
  0.068106  0.087443  0.093242  0.082958  0.070195  0.074170  0.094851  0.106470
  0.087558  0.053072  0.039847  0.060335  0.088045  0.093923  0.080546  0.069747
  0.070331  0.073302  0.071227  0.065995  0.058997  0.046771  0.029195  0.013548
  0.008241  0.052662  0.066490  0.076616  0.076203  0.073069  0.083255  0.106560
```

DATA OMITTED

```
</TARGET_PATTERN>

<NAME> 00001fa010_930831 </NAME>

<WIDTH> 92 </WIDTH>

<HEIGHT> 138 </HEIGHT>

<NUMLAYERS> 10 </NUMLAYERS>

<CONNECTIONS>
0 0 1 1 1 1 1 1 1 0
0 0 1 1 1 1 1 1 1 0
0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 0
</CONNECTIONS>

<TH_INIT>
   1.500000e-001 1.500000e-001 2.500000e+000 2.500000e+000 2.500000e+000
   2.500000e+000 2.500000e+000 2.500000e+000 2.500000e+000 2.500000e+000 50000000
</TH_INIT>

<V_INIT>
0 0 0 0 0 0 0 0 0 0
</V_INIT>
```

```
<S_INIT>
1 1 1 1 1 1 1 1 1 1 9.999000e-001
</S_INIT>

<BIAS>
1 -1 1 1 1 1 -1 -1 -1 -1 1
</BIAS>

<GROUP>
1 1 2 2 2 2 2 2 2 2 4
</GROUP>

<IMAGE_SCALE>
1 1 1 1 1 1 1 1 1 1 1
</IMAGE_SCALE>

<TH_INIT_TARGET> 50000000 </TH_INIT_TARGET>

<V_INIT_TARGET> 0 </V_INIT_TARGET>

<S_INIT_TARGET> 9.999000e-001 </S_INIT_TARGET>

<IMAGE_SCALE_TARGET> 1 </IMAGE_SCALE_TARGET>

<BIAS_TARGET> 1 </BIAS_TARGET>

<NORMALIZE> 2 </NORMALIZE>

<MAXTIME> 500 </MAXTIME>

<CURRENTTIME> 50 </CURRENTTIME>

<CUSTOM>
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0 0 0 0 0 0 0 0
```

```
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 -
   1.315334e-007 -1.747882e-007 -2.481516e-007 -1.828085e-007 -3.219407e-007 -
   3.992974e-007 -4.289843e-007 -3.992974e-007 -3.219407e-007 -1.828085e-007 -
   7.336332e-008 0 0 0 0 0 0 0 1.537752e-007 1.537752e-007 0 0 0 0 0 0 0 0 0 0 0
   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 -9.282684e-008
   -5.842954e-007 -1.456526e-006 -2.217973e-006 -2.382698e-006 -2.145165e-006 -
   2.069002e-006 -2.251694e-006 -2.660539e-006 -2.922818e-006 -2.720865e-006 -
   1.948404e-006 -1.312803e-006 -1.214137e-006 -6.726788e-007 -1.771531e-007
   2.322991e-007 4.147149e-007 4.962998e-007 7.716004e-007 1.037972e-006
   1.451884e-006 1.764388e-006 1.415971e-006 7.480560e-007 1.330702e-007 0 0 0 0 0
   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

DATA OMITTED

```
</CUSTOM>

<IMAGE>
111 106 103 97 93 93 90 89 87 84 82 80 79 79 77 76 76 77 77 75 77 77 77 78 80 79 80
   80 81 80 80 81 82 81 81 82 81 82 82 81 82 80 81 80 79 80 79 78 78 77 76 77 77
   79 80 80 81 82 82 81 80 79 78 76 77 78 82 83 85 85 86 86 86 85 85 86 85 85 85
   84 85 84 85 89 91 94 96 97 100 102 106 113
108 106 100 96 93 92 88 86 85 85 83 81 79 78 77 76 77 76 76 77 77 77 77 78 79 80 80
   79 81 80 80 80 81 82 81 80 81 81 82 81 81 80 81 80 80 79 78 77 76 76 77 77 77
   78 78 78 80 79 80 79 79 78 77 76 77 79 81 83 85 86 87 86 85 83 84 84 84 84 85
   84 85 86 87 89 91 93 97 99 100 101 106 113
```

DATA OMITTED

```
</IMAGE>

<KERNEL>
  <NAME> On </NAME>
  <LAYER> 1 </LAYER>
  <KWIDTH> 5 </KWIDTH>
  <KHEIGHT> 5 </KHEIGHT>
  <TYPE> log </TYPE>
  <DIM> 5 </DIM>
  <SIGMA> 5.000000e-001 </SIGMA>
  <PHASE> -1 </PHASE>
</KERNEL>

<KERNEL>
  <NAME> Off </NAME>
  <LAYER> 2 </LAYER>
  <KWIDTH> 5 </KWIDTH>
  <KHEIGHT> 5 </KHEIGHT>
  <TYPE> log </TYPE>
  <DIM> 5 </DIM>
  <SIGMA> 5.000000e-001 </SIGMA>
  <PHASE> 1 </PHASE>
</KERNEL>
```

```
<KERNEL>
  <NAME> 0 Degrees </NAME>
  <LAYER> 3 </LAYER>
  <KWIDTH> 15 </KWIDTH>
  <KHEIGHT> 15 </KHEIGHT>
  <TYPE> gabor </TYPE>
  <DIM> 15 </DIM>
  <FREQUENCY> 15 </FREQUENCY>
  <ALIGNMENT> 0 </ALIGNMENT>
  <SIGMA> 2.500000e+000 </SIGMA>
  <PHASE> 90 </PHASE>
  <MEAN> 0 </MEAN>
  <AMPLITUDE> 1 </AMPLITUDE>
</KERNEL>

<KERNEL>
  <NAME> 45 Degrees </NAME>
  <LAYER> 4 </LAYER>
  <KWIDTH> 15 </KWIDTH>
  <KHEIGHT> 15 </KHEIGHT>
  <TYPE> gabor </TYPE>
  <DIM> 15 </DIM>
  <FREQUENCY> 15 </FREQUENCY>
  <ALIGNMENT> 45 </ALIGNMENT>
  <SIGMA> 2.500000e+000 </SIGMA>
  <PHASE> 90 </PHASE>
  <MEAN> 0 </MEAN>
  <AMPLITUDE> 1 </AMPLITUDE>
</KERNEL>

<KERNEL>
  <NAME> 90 Degrees </NAME>
  <LAYER> 5 </LAYER>
  <KWIDTH> 15 </KWIDTH>
  <KHEIGHT> 15 </KHEIGHT>
  <TYPE> gabor </TYPE>
  <DIM> 15 </DIM>
  <FREQUENCY> 15 </FREQUENCY>
  <ALIGNMENT> 90 </ALIGNMENT>
  <SIGMA> 2.500000e+000 </SIGMA>
  <PHASE> 90 </PHASE>
  <MEAN> 0 </MEAN>
  <AMPLITUDE> 1 </AMPLITUDE>
</KERNEL>

<KERNEL>
  <NAME> 135 Degrees </NAME>
  <LAYER> 6 </LAYER>
  <KWIDTH> 15 </KWIDTH>
  <KHEIGHT> 15 </KHEIGHT>
  <TYPE> gabor </TYPE>
  <DIM> 15 </DIM>
  <FREQUENCY> 15 </FREQUENCY>
  <ALIGNMENT> 135 </ALIGNMENT>
  <SIGMA> 2.500000e+000 </SIGMA>
  <PHASE> 90 </PHASE>
  <MEAN> 0 </MEAN>
  <AMPLITUDE> 1 </AMPLITUDE>
</KERNEL>
```

```
<KERNEL>
  <NAME> 180 Degrees </NAME>
  <LAYER> 7 </LAYER>
  <KWIDTH> 15 </KWIDTH>
  <KHEIGHT> 15 </KHEIGHT>
  <TYPE> gabor </TYPE>
  <DIM> 15 </DIM>
  <FREQUENCY> 15 </FREQUENCY>
  <ALIGNMENT> 180 </ALIGNMENT>
  <SIGMA> 2.500000e+000 </SIGMA>
  <PHASE> 90 </PHASE>
  <MEAN> 0 </MEAN>
  <AMPLITUDE> 1 </AMPLITUDE>
</KERNEL>

<KERNEL>
  <NAME> 225 Degrees </NAME>
  <LAYER> 8 </LAYER>
  <KWIDTH> 15 </KWIDTH>
  <KHEIGHT> 15 </KHEIGHT>
  <TYPE> gabor </TYPE>
  <DIM> 15 </DIM>
  <FREQUENCY> 15 </FREQUENCY>
  <ALIGNMENT> 225 </ALIGNMENT>
  <SIGMA> 2.500000e+000 </SIGMA>
  <PHASE> 90 </PHASE>
  <MEAN> 0 </MEAN>
  <AMPLITUDE> 1 </AMPLITUDE>
</KERNEL>

<KERNEL>
  <NAME> 270 Degrees </NAME>
  <LAYER> 9 </LAYER>
  <KWIDTH> 15 </KWIDTH>
  <KHEIGHT> 15 </KHEIGHT>
  <TYPE> gabor </TYPE>
  <DIM> 15 </DIM>
  <FREQUENCY> 15 </FREQUENCY>
  <ALIGNMENT> 270 </ALIGNMENT>
  <SIGMA> 2.500000e+000 </SIGMA>
  <PHASE> 90 </PHASE>
  <MEAN> 0 </MEAN>
  <AMPLITUDE> 1 </AMPLITUDE>
</KERNEL>

<KERNEL>
  <NAME> 315 Degrees </NAME>
  <LAYER> 10 </LAYER>
  <KWIDTH> 15 </KWIDTH>
  <KHEIGHT> 15 </KHEIGHT>
  <TYPE> gabor </TYPE>
  <DIM> 15 </DIM>
  <FREQUENCY> 15 </FREQUENCY>
  <ALIGNMENT> 315 </ALIGNMENT>
  <SIGMA> 2.500000e+000 </SIGMA>
  <PHASE> 90 </PHASE>
  <MEAN> 0 </MEAN>
  <AMPLITUDE> 1 </AMPLITUDE>
</KERNEL>

</TARGET>

</ROOT>
```

Distribution

1	MS	0974	Larry J. Ellis, 5502
1		1170	Russell, D. Skocypec, 15310
1		0451	David P. Gallegos, 5533
1		1188	John S. Wagner, 15311
15		0451	Kurt W. Larson, 5533
15		0451	Paul C. Reeves, 5533
6		0451	Antonio I. Gonzales, 5533
6		0451	Benjamin D. Farkas, 5533
6		0451	John J. Jones, 5533
1		0451	John H. Ganter, 5533
1		1188	James C. Forsythe, 15311
1		0513	Pace J. VanDevender, 1000
1		1231	Alton D. Romig, 5000
1		1451	Samuel G. Varnado, 5500
1		0310	Alejandro Bäcker, 9212
1		1003	John T. Feddema, 15211
1		1188	Ann E. Speed, 12335
1		0321	William J. Camp, 9200
1		1004	Raymond W. Harrigan, 15221
1		0321	Robert W. Leland, 9220
6		0784	Ronald E. Trellue, 5501
1		0974	William J. Slosarik, 5520
1		0511	Charles E. Meyers, 1010
1		0323	Henry R. Westrich, 1011
1		0323	Keith Ortiz, 1011
1		0310	Mark D. Rintoul, 9212
1		9018	Central Technical Files, 8940-2
5		0899	Technical Library, 4916
2		0619	Review & Approval Desk, 12690 For DOE/OSTI